PRACTICA TU

BASIC

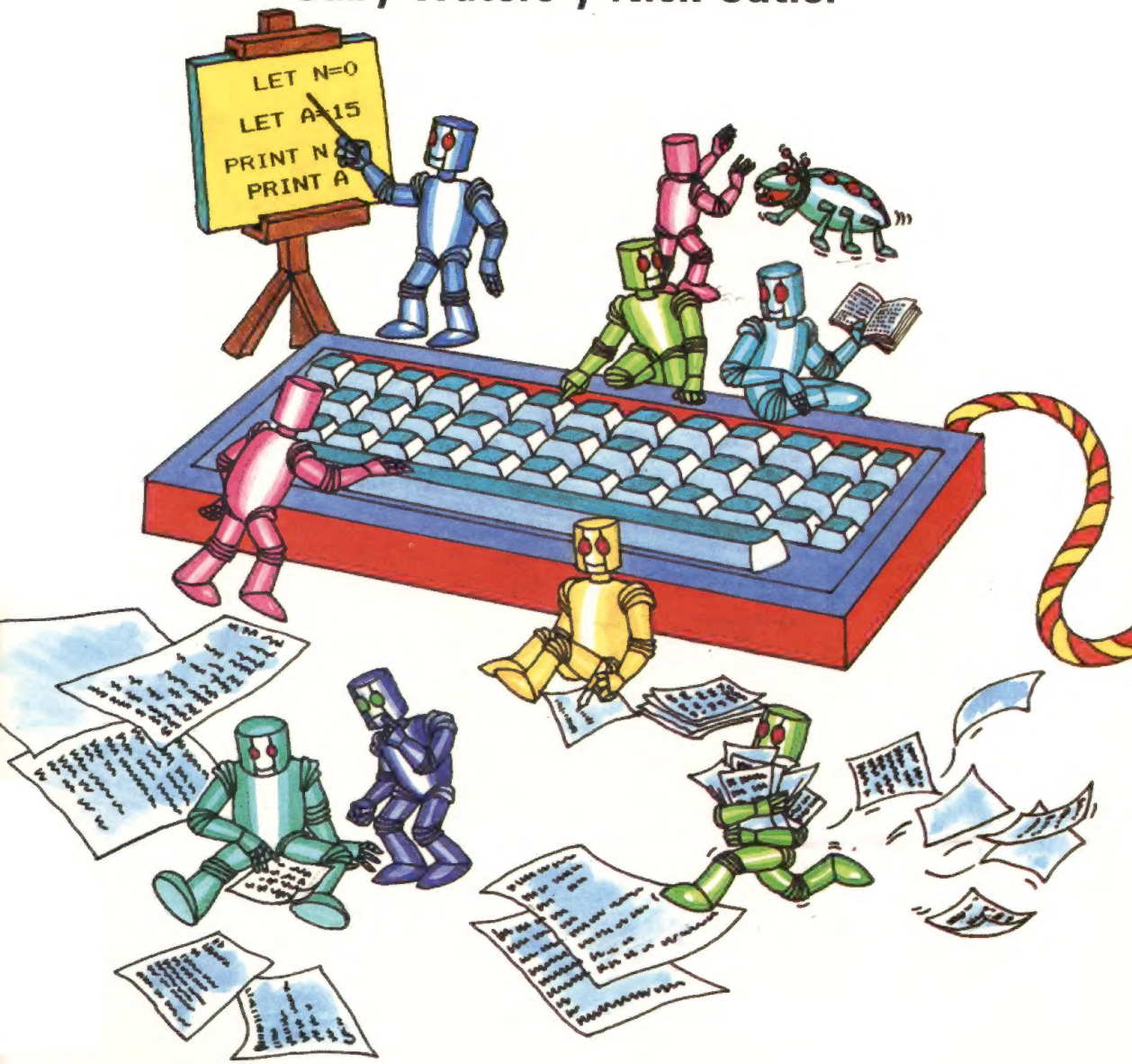
Puzzles, ejercicios y problemas

Ediciones Plesa

Colección Electrónica

PRACTICA TU BASIC

Gaby Waters y Nick Cutler

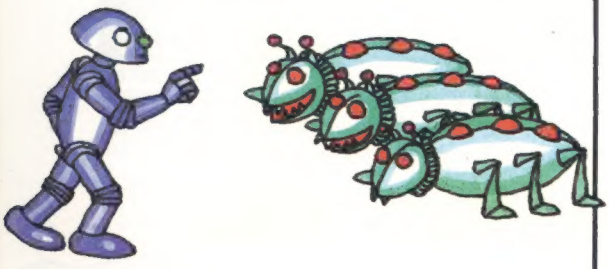


Contenido

Sobre este libro

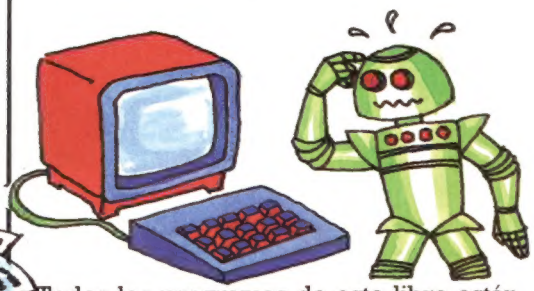
Este libro contiene un gran número de ejercicios y problemas para resolver escribiendo programas que te ayudarán a practicar tu BASIC. Hay programas a los que les faltan pasos y variables que introducir, listados de programas llenos de errores para encontrar y corregir e ideas para programas que puedes escribir. El libro trata todas las instrucciones BASIC importantes, empezando con sencillas sentencias PRINT y terminando con consejos para escribir un programa largo de un juego de búsqueda del tesoro.

Los programas llenos de errores y defectos están marcados con un «chinche» como el que está al principio de este párrafo. Esto te indica que debes buscar los errores y corregirlos para que el programa funcione correctamente. Otros programas tienen pasos y variables para rellenar. Los pasos que tienes que completar están marcados con un asterisco y los espacios a rellenar están marcados con interrogaciones.



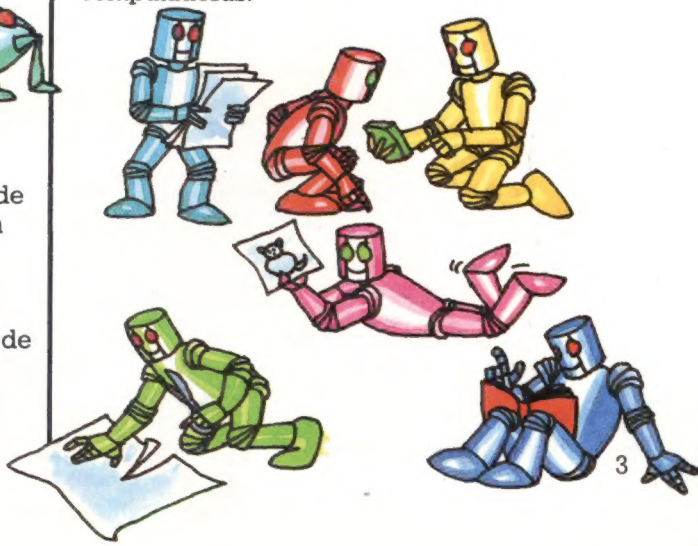
Las respuestas a todos los problemas de programas se dan al final del libro con explicaciones detalladas. Usa las respuestas cuando quedes atascado especialmente en los programas más largos. Frecuentemente las respuestas de un problema en los primeros pasos puede ayudarte a resolver problemas posteriores o darte una idea para escribir el resto del programa solo.

Algunas veces los pasos que encuentres en las soluciones diferirán de los que tú has escrito. Si tu programa funciona perfectamente, no te preocupes, a veces hay dos formas de escribir el mismo programa. Prueba a comparar tus respuestas con las del libro para ver qué método es el más efectivo.



Todos los programas de este libro están escritos en un BASIC standard, lo que significa que deberían funcionar en la mayoría de las computadoras sin mucha alteración. Algunas palabras son diferentes en todas las computadoras, así que si escribes un programa y obtienes un error debes revisar las palabras BASIC del programa. Hay una tabla de conversiones en la página 47 para ayudarte.

Si no estás seguro de lo que hace una instrucción BASIC, *hay una guía sobre* BASIC que contiene todas las palabras usadas en este libro y una breve explicación de cada una. Se dan consejos especiales para escribir programas en computadoras Sinclair (Timex) y también hay respuestas alternativas para estas computadoras.

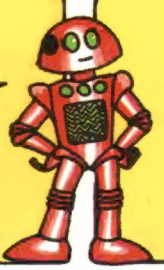


Conociendo el BASIC

Estas dos páginas te darán algo de práctica en la utilización de la instrucción PRINT. Esta le dice a la computadora que escriba algo en la pantalla. Puedes usar PRINT como una instrucción directa, esto es, sin número de paso de programa, y la computadora ejecutará la instrucción directamente. Después de una instrucción directa tienes que presionar RETURN o ENTER o NEWLINE, según la computadora.

```
PRINT «PEZ»
PRINT «2 PECES»
PRINT 2345
```

Recuerda presionar RETURN o la palabra de tu computadora después de cada instrucción.



```
PRINT 2+2+3
PRINT 6*8
PRINT 15-4
PRINT 16/4
PRINT SQR(16)
PRINT 5346-257
```

Prueba a teclear estas instrucciones directas en tu computadora. Cuando le dices a la computadora que imprima letras, o letras y números juntos, éstas tienen que estar entre comillas. Los números por sí solos no necesitan comillas.

También puedes usar PRINT para hacer que la computadora haga cálculos. Aquí hay algunas operaciones sencillas con símbolos matemáticos en BASIC. Si no estás seguro de lo que significan, ejecútalos.

```
PRINT «HOLA», «HOLA»
PRINT «HOLA»; «HOLA»
PRINT «HOLA»; «HOLA»
PRINT 555,777
```

Todas estas instrucciones le dicen a la computadora dónde escribir en la pantalla. Pruébalas para ver qué hacen. Los signos de puntuación tienen significados especiales en BASIC. Una coma le dice a la computadora

```
PRINT «     HOLA»
PRINT SPC(10) «HOLA»
PRINT TAB(15); «HOLA»
PRINT TAB(10); 555
```

que deje algunos espacios antes de imprimir el siguiente elemento y un punto y una coma le dice que imprima el siguiente elemento en la misma línea sin dejar ningún espacio.

Corrección de errores y alteración de programas.

Los errores en los programas se llaman bugs (chinches). Pueden estar causados por simples errores de escritura o por romper las reglas del BASIC. Tendrás que averiguar cómo corregir errores en la computadora y cómo alterar programas.

HOLA

La mayor parte de las computadoras tienen una tecla DELETE o RUBOUT (borrar) para corregir fallos.

Para insertar una letra que falte tienes que mover el cursor. Mira en tu manual para averiguar cómo se usan las teclas de control del cursor.

Para borrar un paso entero escribe el número de paso seguido de RETURN (o la palabra de tu computadora).

Problemas de programas sencillos

Aquí hay un programa para probar. Hace que la computadora exponga el mensaje de la pantalla de la derecha. Prueba a ejecutar el programa y después intenta cambiarlo para que la pantalla se parezca a las que se muestran abajo. Las instrucciones directas de la página opuesta deberían ayudarte. Hay algunas ideas sobre alteración de programas y corrección de errores en la parte de abajo de la página.

```
10 PRINT «¿CUAL ES TU NOMBRE?»
20 INPUT A$
30 PRINT
40 PRINT «HOLA»
50 PRINT A$
```

Hace que la computadora espere a que teclees tu nombre y entonces lo almacena en una variable llamada A$.

PRINT por sí solo produce una línea en blanco.

Dice a la computadora que escriba la palabra almacenada en A$.

```
¿CUAL ES TU NOMBRE?
?ROBOT

HOLA
ROBOT
```

Teclea el programa en tu computadora presionando RETURN (o la palabra de tu computadora) después de cada paso. Al final teclea RUN y presiona RETURN.

1
```
¿CUAL ES TU NOMBRE?
?ROBOT

HOLA ROBOT
```

2
```
¿CUAL ES TU NOMBRE?
?ROBOT

HOLA        ROBOT
```

Prueba a alterar el programa para hacer que la pantalla se parezca a éstas.

3
```
¿CUAL ES TU NOMBRE?
?ROBOT

HOLA ROBOT
```

4
```
¿CUAL ES TU NOMBRE?
?ROBOT

ADIOS
ROBOT
```

La diferencia entre el programa y las instrucciones directas en la página opuesta es que cada línea de instrucciones en un programa tiene un número. La computadora almacena las instrucciones en su memoria y no las lleva a cabo hasta que teclees RUN. Si usas números de líneas que vayan de 10 en 10, puedes añadir instrucciones adicionales sin renumerar el programa.

Para ver el listado del programa en la pantalla, teclea LIST.

Para borrar un programa entero, teclea NEW.

```
10 PRINT «ESTE ES UN»
20 PRINT «PROGRAMA»
30 PRINT «ESTUPIDO»
```

Puedes añadir nuevos pasos en cualquier parte del programa o corregir los antiguos tecleándolos de nuevo. Prueba a teclear este programa tal cómo esta aquí y observa lo que pasa cuando lo ejecutas o sacas del listado.

Problemas de variables

Una variable es un espacio en la memoria de la computadora con un nombre, donde se puede almacenar información. Cuando introduzcas información en la computadora, utiliza las palabras LET o INPUT. La información que contenga palabras o una mezcla de letras, números y símbolos se llama cadena. Una cadena debe estar entre comillas y su nombre tiene un signo dólar detrás.

```
10 LET A=16
20 LET R$=«ROBOTS OXIDADOS»
30 PRINT A
40 PRINT R$

RUN
16
ROBOTS OXIDADOS
```

A y R$ son los nombres de variables.

```
10 PRINT «¿CUAL ES TU NOMBRE?»
20 INPUT N$
30 PRINT «¿CUANTOS AÑOS TIENES?»
40 INPUT A
50 PRINT N$; « ES »;A
```

Variable de cadena

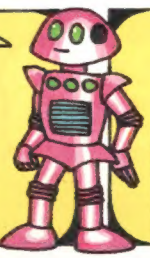
LET le dice la computadora que ponga nombre a un espacio de la memoria y que ponga alguna información en él.

INPUT indica un espacio en la memoria y hace que la computadora espere a que teclees la información cuando ejecutas el programa.

Elección de nombres de variables

¿Cuáles de éstas son legales en tu computadora?

```
LET B$=«6422 RATAS»
LET B5=6422
LET K1=99
LET MO$=«MOSCA»
```



Algunos de estos nombres de variables contienen palabras en BASIC y producirán un error. ¿cuáles son?

```
LET LETRA$=«HI!»
LET DIEZ=10
LET RUN$=«MAS RATAS»
LET MO$=«50 MOSCAS»
```

El BASIC es un quisquilloso sobrenombre que asignas a las variables y sus reglas varían en diferentes computadoras. Por ejemplo, el ZX81 (Sinclair 1000) sólo acepta nombres de una letra para variables de cadena.*

Si usas palabras cortas como nombres de variables no debes usar aquellas que contengan palabras de BASIC, pues confundirán a la computadora. Prueba los pasos dados arriba para ver cuáles son legales en tu computadora.

impresión de palabras y variables

```
10 LET RR$=«ROBOT OXIDADO»
20 PRINT «HOLA» »; RR$
```

Prueba al teclear una coma en vez de un punto y coma y observa qué ocurre.

```
10 LET R$=«OXIDADO»
20 LET S=66
30 PRINT R$;» LOS ROBOTS COMIERON »
40 PRINT S;« SALCHICHAS »
```

Cuando escribas palabras y variables juntas, las palabras deben ir entre comillas y debes poner punto y coma entre ellas y la variable. Tendrás que dejar un espacio dentro de las comillas en ambos lados de las palabras. Prueba a quitar los espacios en los pasos de arriba y observa lo que sucede.

Problema con PRINT

```
LET A=66
LET B=77
LET RR$=«ROBOT OXIDADOS»
```

```
66 ROBOTS OXIDADOS COMIERON
77 SALCHICHAS CHAMUSCADAS
```

¿Puedes escribir un programa usando las variables que se dan arriba de manera que cuando lo ejecutes su pantalla se parezca a ésta?

* Si tienes un ZX81 tendrás que alterar los nombres de las variables de cadena de más de una letra cuando aparezcan en este libro.

Fíjate en un programa

Este programa convierte temperaturas Fahrenheit a grados centígrados. Si no estás seguro de cómo funciona el programa o de lo que hacen las variables, prueba a añadir sentencias PRINT para que la computadora escriba el contenido de las variables en la pantalla. De esta manera podrás mirar «dentro» del programa y entender cómo funciona.

```
10 INPUT A
20 LET F=32
30 LET B=5
40 LET C=B/9
50 LET D=A-F
60 LET R=D*C
70 PRINT R
```

La temperatura que introduces se almacena en la variable A.

Intenta añadir algunos pasos al programa para averiguar los valores de D y C.

Problema: Alterar un programa

Ahora intenta cambiar este programa para convertir las temperaturas de grados centígrados a grados Fahrenheit.

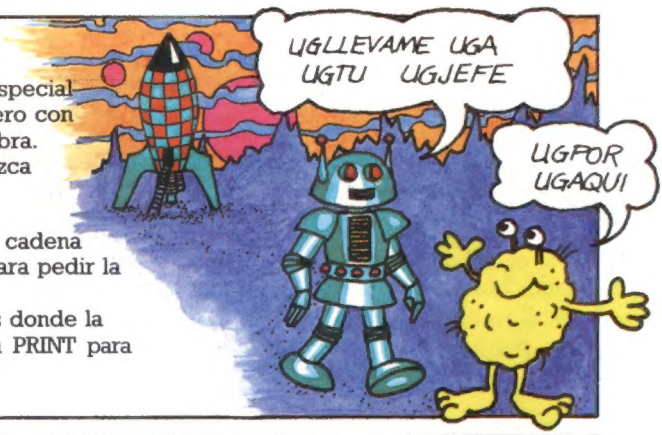
Problema: Escribir un programa

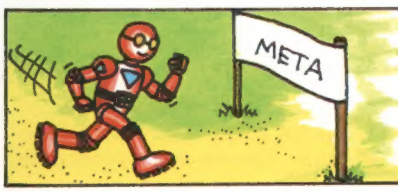
En los recuadros inferiores hay algunas ideas de programas sencillos para escribir. Tendrás que usar muchas variables, así es que anótalas con los nombres que escojas. Es útil elegir nombres de variables que te recuerden lo que significan o para lo que sirven . Primero intenta escribir los programas en papel, a continuación pásalos a la computadora y corrígelos si es necesario.

1. ¿Hablas Ugiano?

Te ha sido encomendada una misión especial con los Ugianos que hablan español pero con las letras UG al principio de cada palabra. Intenta escribir un programa que traduzca palabras españolas a Ugiano.

Pistas: Utiliza LET con una variable de cadena para contener las letras UG e INPUT para pedir la palabra en español. Luego haz que la computadora exponga ambas variables donde la palabra sea en Ugiano. Utiliza pasos con PRINT para que resulte más claro el programa.





2. Cálculo de la velocidad

¿Sabrías escribir un programa que calculase la velocidad? Necesitarás pasos INPUT para el tiempo y la distancia, una operación para calcular la velocidad y pasos PRINT para hacer más claro el programa.

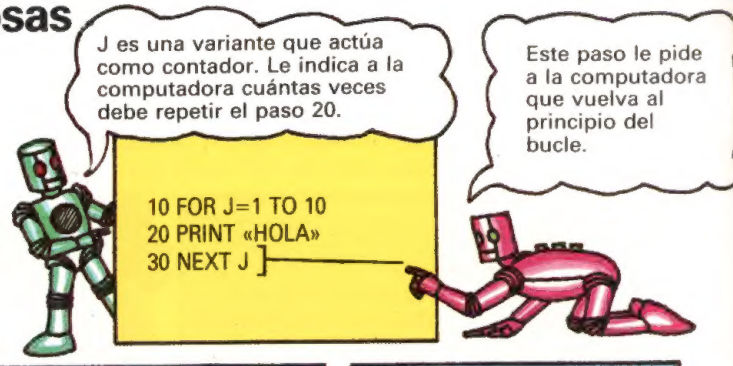
Programa sobre salchichas

El Robot 1 come 30 salchichas por hora mientras el Robot 2 come sólo 20 por hora. Si el Robot 2 quiere comer 35 salchichas y el Robot 1 se niega a comer más despacio, ¿cuántas salchichas tendrán que comprar y cuánto tiempo tardarán en comérselas todas?

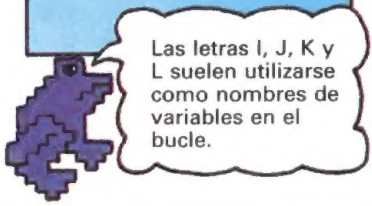
Pistas: Comienza con tres variables, dos para contener el número de salchichas que pueden comer los robots y una tercera que contenga el número que desea comer el Robot 2. Usa más variables para contener el tiempo que el Robot 2 tarda en comer 35 salchichas y el número de ellas que el Robot 1 puede comer en ese tiempo.

Repetición de cosas

Es muy útil saber hacer que la computadora repita algo varias veces. Una manera de hacer esto es mediante un bucle con las palabras FOR, TO y NEXT. Ejecuta el programa de la derecha para ver cómo funciona un bucle.




```
10 FOR K=1 TO 10
20 PRINT K
30 NEXT K
```




```
10 FOR I=1 TO 10
20 PRINT I;« X 8»;
30 PRINT «= »;I*8
40 NEXT I
```

```
10 FOR L=1 TO 15
20 PRINT TAB(L);
«HOLA»
30 NEXT L
```

Puedes ver como funciona una variable de bucle introduciendo un paso PRINT que muestre en la pantalla su valor. También puedes usar el valor de la variable dentro del bucle. Prueba éstos.

Bucles

1

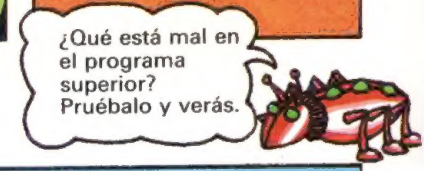
```
HOLA HOLA HOLA H
OLA HOLA HOLA HO
LA HOLA HOLA HOL
A HOLA HOLA HOLA
```

2

```
HOLA
HOLA
HOLA
HOLA
HOLA
HOLA
```

3

```
10 FOR I=1 TO 20
20 LET I=I-1
30 PRINT I
40 NEXT I
```



¿Se te ocurre un programa simple con un bucle que llene la pantalla con HOLAS y otro que muestre una columna de HOLAS en el centro de la pantalla?

Step (saltos)

```
10 FOR I=1 TO 25 STEP 5
20 PRINT I
30 NEXT I
```

```
10 FOR J=20 TO 1 TO STEP-1
20 PRINT J
30 NEXT J
```

Puedes cambiar la manera de contar de la variable utilizando la palabra STEP (saltar) y un número que indica a la computadora que debe contar dando los saltos que ese número indica. Si introduces un número negativo, la computadora contará hacia atrás. Ejecuta estos programas; luego vuelve a ejecutarlos pero variados los números de STEP.

1

```
HOLA
 HOLA
  HOLA
   HOLA
    HOLA
     HOLA
```

2

5	25
4	16
3	9
2	4
1	1
0	0

3

```
COMENZAR ?12
PARAR ?0
SALTAR ?-3
12  9  6  3  0
```

¿Sabrías escribir programas para que la pantalla saliese así?

Bucles de retardamiento

Las computadoras funcionan a diferentes velocidades, por lo que quizá tengas que hacer el bucle mayor o menor alternando la cifra 1000.

```
10 FOR J=10 TO 1 STEP-1
20 PRINT J
30 NEXT J
40 FOR K=1 TO 1000
50 NEXT K
60 PRINT «SUELTA YA»
```

Un bucle de retardamiento es un bucle vacío son ningún tipo de instrucciones. En este programa los pasos 40-50 hacen que la computadora cuente del 1 al 1000, lo que significa que no hace nada durante ese tiempo.

```
MENSAJE SECRETO
MEMORIZA EN 5 SEGUNDOS
LUEGO DESAPARECERA

REUNIRSE CON AGENTE X 2.00 AEROPUERTO
```

Intenta escribir un programa para que la computadora muestre un mensaje secreto como el que se muestra aquí, para hacerlo desaparecer a los 5 segundos. Para hacer esto necesitas un bucle de retadamiento seguido de la correspondiente instrucción de borrado de tu computadora. Prueba varias veces para calcular la cifra exacta que debes colocar en el bucle.

Programa: Dibujo

1

```
10 LET A$=«***»
20 FOR J=1 TO 7
30 PRINT TAB(J);A$
40 NEXT J
50 FOR K=1 TO 3
60 PRINT TAB(J+1);A$
70 NEXT K
80 FOR L=7 TO 1 STEP-1
90 PRINT TAB(L);A$
100 NEXT L
```

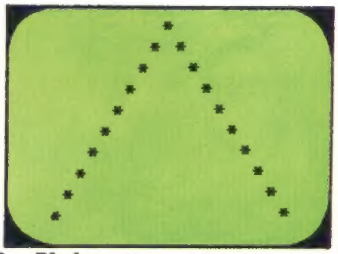
Puedes utilizar caracteres gráficos en lugar de estrellas, siempre que tu computadora los tenga.

```
80 FOR L=8 TO 18
90 PRINT TAB(L);A$;«**»
100 NEXT L
```

El programa de la izquierda realiza un dibujo con estrellas en la pantalla. Ejecútalo y después prueba a alterarlo para realizar diferentes dibujos. Encima tienes una idea.

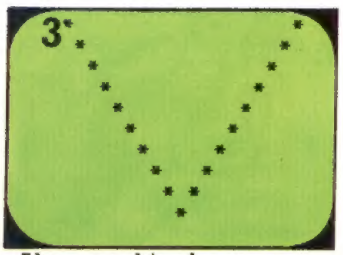
2

```
10 CLS
20 LET A=15
30 PRINT TAB(A);«*»
40 FOR K=1 TO .9
* 50 PRINT TAB(?);«*«;
* 60 PRINT TAB(?);»*»
* 70 ?
```



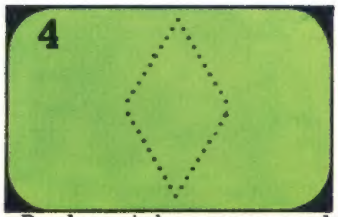
¿Sabrías completar los pasos 50 a 70 de este programa para que tu pantalla aparezca como la del dibujo?

3*



Ahora cambia el programa para que la figura salga al revés.

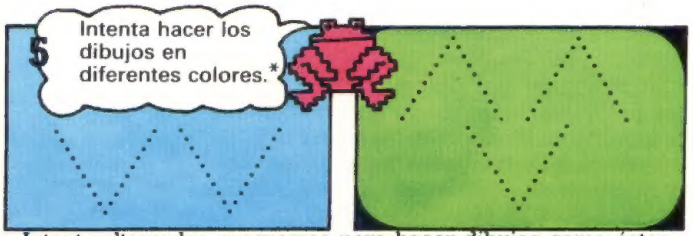
4



Puedes unir los programas de los problemas 2 y 3 para realizar la figura de un diamante.

5

Intenta hacer los dibujos en diferentes colores.*



Intenta alterar los programas para hacer dibujos como éstos.

* Busca las instrucciones de colores de tu computadora en el manual.

Problemas de bucles

Puedes usar bucles dentro de bucles. Estos se conocen como bucles anidados. Cada vez que se repite el bucle externo, el bucle interno se repetirá un número determinado de veces.

```
10 FOR K=1 TO 3
20 FOR L=1 TO 5
30 PRINT K,L
40 NEXT L
50 NEXT K
```

Bucle anidado

Este programa escribe el valor de las variables del bucle para que puedas ver cómo funcionan los bucles anidados.

```
10 FOR I=1 TO 4
20 FOR J=1 TO 4
30 PRINT
40 NEXT J
50 PRINT «HOLA»
60 NEXT I
```

Ejecuta este programa y a continuación prueba a cambiar el tamaño de los bucles. ¿Sabrías incluir otro bucle aninado para que el programa fuese más lento?

Chinches en los bucles

```
10 FOR L=1 TO 15
20 PRINT TAB(5);«*»;
TAB (10);«*»
30 FOR J=1 TO 5
40 PRINT
50 NEXT J
60 NEXT L
```

```
10 FOR I=9  TO 0 STEP-1
20 FOR J=9  TO 0 STEP-1
30 FOR K=9  TO 0 STEP-1
40 PRINT I; J; K
50 NEXT I
60 NEXT J
70 NEXT K
```

Debes tener cuidado de incluir ambos componentes del bucle dentro del otro bucle, ya que si no obtendrás un chinche (error). El programa de la izquierda es correcto pero no el de la derecha; intenta localizar los errores que tiene.

Contador binario

```
10 FOR A=0 TO 1
20 FOR B=0 YO 1
30 FOR C=0 TO 1
40 FOR D=0 TO 1
50 PRINT D+C*2+B*4+A*8;« = »;
60 PRINT A;B;C;D
70 NEXT D
80 NEXT C
90 NEXT B
100 NEXT A
```

El paso 50 calcula el valor decimal de cada dígito binario.

La computadora repite primero el bucle más interno.

Este programa utiliza cuatro bucles anidados para contar en binario. Puede contar muy deprisa, ya que sólo utiliza números binarios de cuatro cifras. ¿Sabrías añadir más bucles y cambiar los pasos PRINT para crear un programa que cuente con números binarios de ocho cifras?

Mensaje intermitente

¿Sabrías escribir un programa que escribiese un mensaje como éste que estuviese intermitente? Necesitarás un bucle para borrar la pantalla y escribir el mensaje un número determinado de veces, y dos bucles de retardamiento, para que el mensaje no aparezca y desaparezca demasiado rápidamente.

Reloj de la computadora

Utiliza un reloj real para establecer la duración del bucle en un segundo.

Intenta escribir un programa para que tu computadora funcione como un reloj. Necesitarás un bucle para los segundos, otro para los minutos y un bucle de retardamiento.

Programa para hacer despegar un cohete

¿Sabrías escribir un programa usando varios bucles que represente un cohete sencillo «despegando» y moviéndose lentamente por la pantalla?

1. Tendrás que borrar la pantalla y escribir montones de pasos vacíos para lograr que el cohete aparezca en la parte inferior de la pantalla.

```
  *
 ***
 ***
 ***
** **
```

Puedes utilizar caracteres de gráficos si la computadora te lo permite.

```
  *
 ***
 ***
 ***
** **
```

2. Puedes realizar el dibujo de un cohete utilizando caracteres normales en sentencias PRINT.

3. Para hacer que el cohete se mueva, añade más pasos PRINT vacíos y un bucle de retardamiento para que no se mueva demasiado rápido.

Programa del saltador

Intenta escribir un programa de animación sencillo que represente una figura como la de la derecha que salte de arriba a abajo según avanza por la pantalla.

```
<0>   0     <0>   0     <0>   0
 0   <0>    0   <0>    0   <0>
/\    /\    /\    /\    /\    /\
```

Dibujar la figura

Puedes dibujar una figura sencilla en dos posiciones como las que se muestran a la derecha, utilizando PRINT con letras y símbolos.

```
PRINT «<0>»
PRINT «0»
PRINT «/  \»
```

Añade los números de paso que quieras.

Necesitarás este paso vacío para que parezca que la figura salta.

```
PRINT
PRINT «0»
PRINT «<0>»
PRINT «/  \»
```

Escribir el programa

1. Necesitarás un bucle con el tamaño de la anchura de tu pantalla. Debería contar de dos en dos. Cuando tengas que escribir las instrucciones PRINT TAB entenderás por qué.

2. Dentro del bucle tendrás que borrar la pantalla y a continuación representar la figura. *Utiliza TAB para que la posición de la figura cambie cada vez que se represente.*

3. Añade un bucle de retardamiento para que la figura permanezca un breve período en la pantalla.

4. Repite esta rutina con una instrucción TAB diferente para la segunda figura.

```
<0>   0     <0>   0     <0>   0     <0>   0     <0>   0     <0>   0     <0>
 0   <0>    0   <0>    0   <0>    0   <0>    0   <0>    0   <0>    0   <0>
/\    /\    /\    /\    /\    /\    /\    /\    /\    /\    /\    /\    /\
```

Ejercicios con IF/THEN

Las palabras IF (SI) y THEN (ENTONCES) se utiliza para comparar información y que la computadora haga cosas diferentes según los resultados.

>=significa mayor o igual que
<=significa menor o igual que.

```
10 LET A=8
20 INPUT B
30 IF B=A THEN PRINT «SON IGUALES»
40 IF B<>A THEN PRINT «NO SON IGUALES»
50 IF B<A THEN PRINT B; «ES MENOR QUE»; A
60 IF B>A THEN PRINT B; «ES MAYOR QUE»; A
```

Ejecuta este programa para estar seguro de que entiendes los signos que utiliza la computadora para comparar datos.

```
10 PRINT «¿CUANTOS CHIPS PUEDE COMER?»
20 INPUT C
30 IF C>=0 AND C<=10 THEN PRINT «TE MORIRAS DE
HAMBRE»
40 IF C>10 AND C<=25 THEN PRINT «NO SON
MUCHOS»
50 IF C>25 AND C<=1000 THEN PRINT «GOLOSO»
60 IF C<0 OR C>1000 THEN PRINT «!!!»
```

Casi todas las computadoras te permiten comparar varias cosas al mismo tiempo utilizando las palabras AND (Y) y OR (O) en las sentencias IF/THEN. Ejecuta este programa para ver cómo funcionan.

Examinador de las tablas de multiplicar

```
10 LET A=13
20 FOR J=1 TO 13
30 PRINT «¿CUANTO ES?»; J; «X»; A; «  »;
40 INPUT B
50 IF B=J*A THEN PRINT «CORRECTO»
60 NEXT J
```

Este programa sirve para que se revisen las tablas de multiplicar. ¿Sabrías añadir más instrucciones IF/THEN que te indiquen que la respuesta está mal y cuál es la solución correcta?

Contraseña

```
10 LET S$=«SALCHICHAS»
20 PRINT «LA CONTRASEÑA, POR FAVOR»;
30 INPUT P$
40 IF P$=S$ THEN PRINT «OK, CONTINUA»
```

¿Sabrías completar este programa para que la computadora escriba un mensaje si le das una contraseña equivocada? Añádele también unos pasos para que la computadora pregunte por un número secreto.

Bifurcación

Después de la palabra THEN puedes colocar gran variedad de instrucciones diferentes. Por ejemplo, puedes decirle a la computadora que dé por terminado el programa o bien que salte a otro paso utilizando GOTO.

```
10 LET C=0
20 PRINT «¿ESTAS YA CANSADO?»
30 INPUT B$
40 LET C=C+1
50 IF B$=«SI» THEN STOP
60 IF C>10 THEN PRINT «DEBES ESTARLO»
70 IF B$=«NO» THEN GOTO 20
80 PRINT «NO SEAS TONTO»
90 GOTO 20
```

¿Qué sucede si contestas «plátanos»?

En este programa, la computadora hace diferentes cosas dependiendo de la respuesta que introducirás en el paso 30. Ejecuta el programa varias veces utilizando respuestas diferentes.

Calculadora en la computadora

```
¿DIME UN NUMERO? 7
¿Y OTRO? 11
¿QUE QUIERES: SUMAR,
RESTAR, DIVIDIR O
MULTIPLICAR?
?MULTIPLICAR
LA RESPUESTA ES 77
```

Esta es la muestra de un programa que puede sumar, restar, dividir o multiplicar dos números que tú introduzcas. Intenta escribir el programa y una vez logrado haz cambios para que realice operaciones diferentes.

1

```
10 INPUT X
20 CLS
30 PRINT «ACIERTA EL
   NUMERO»
40 INPUT Y
50 IF Y=X THEN GOTO
   70
60 GOTO 30
70 PRINT «CORRECTO»
```

1. Haz que la computadora te diga si el número que has introducido es demasiado grande o demasiado pequeño. Así te será más fácil adivinarlo.

2. Añade una variable para llevar la cuenta de los intentos que llevas, fija un número límite mediante un paso IF/THEN.

Este programa es un juego para adivinar un número. Una persona escoge un número, la otra tiene que intentar adivinarlo. Intenta mejorar el programa con las sugerencias que se dan arriba.

2

```
PISTA
ANFIBIO
¿RENACUAJO?
NO
¿RANA?
SI
```

Esto representa una pantalla con un juego para adivinar una palabra. ¿Sabrías escribir el programa? Es muy semejante al juego de los números sólo que utiliza variables de cadena.

Carrera de caballos

Este es un listado para un juego de carrera de caballos... pero está incompleto. En los pasos señalados con un asterisco intenta rellenar los números de paso detrás de las instrucciones GOTO. Debajo encontrarás cómo jugar y varias formas para mejorarlo.

En muchas computadoras puedes incluir mensajes en los pasos INPUT. Como en los pasos 20 y 30.

```
 10 LET N=0
 20 INPUT «GANADOR:»; H1
 30 INPUT «SEGUNDO:»; H2
 40 CLS
 50 LET N=N+1
 60 INPUT «ADIVINA EL GANADOR:»; G1
 70 INPUT «ADIVINA EL SEGUNDO:»; G2
*80 IF G1=H1 AND G2=H2 THEN GOTO?
*90 IF G1=H2 OR G2=H1 THEN GOTO?
*100 IF (G1=H1 AND G2<>H2) OR
 (G2=H2 AND G1<>H1) THEN GOTO?
 110 PRINT »MAL»
*120 IF N=4 THEN GOTO?

 130 PRINT «VUELVE A INTENTARLO»: GOTO 50
 140 PRINT «UNO CORRECTO»: GOTO 120
 150 PRINT «CABALLO CORRECTO, POSICION ERRONEA»: GOTO 120
 160 PRINT «CORRECTO»
 170 PRINT «GANADOR:»; H1; «SEGUNDO:»; H2
```

El primer jugador decide qué caballos entran en primera y segunda posición.

N cuenta el número de intentos que realiza el segundo jugador.

Acuérdate de incluir los números de paso.

Muchas computadoras aceptarán dos instrucciones en un paso. Si tu computadora no los acepta, coloca la segunda instrucción en un nuevo paso y omite los dos puntos.

Cómo jugar

Hay seis caballos numerados del 1 al 6. El primer jugador escoge qué caballos llegan en primer y segundo lugar. El segundo jugador tiene cuatro intentos para adivinarlos.

Ideas para mejorar el juego

1. Intenta un sistema de puntuación e introdúcelo en el programa.

2. Ofrece a los jugadores la posibilidad de volver a ejecutarlo.

Números aleatorios

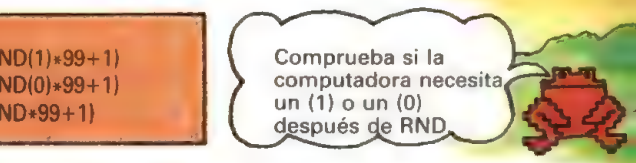
La palabra RND produce un número aleatorio, aunque la manera de utilizarlo varía de una computadora a otra. A continuación te ofrecemos las diferentes instrucciones, pero debes comprobar en el manual de tu computadora cuál es la que te sirve. También puedes mirarlo en la tabla de conversión que hay al final del libro. Después intenta realizar los ejercicios y problemas que se exponen en estas dos páginas.

```
PRINT RND(99)
PRINT RND(10)
```

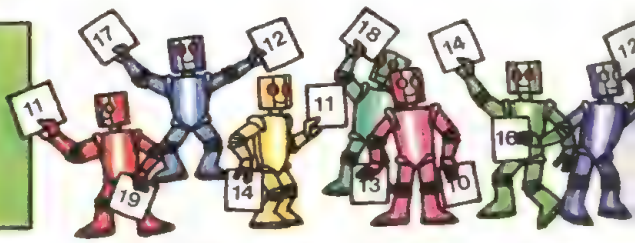
En algunas computadoras utilizas RND con un número entre paréntesis para producir un número entre el uno y el número que pusiste entre paréntesis.

```
PRINT INT(RND(1)*99+1)
PRINT INT(RND(0)*99+1)
PRINT INT(RND*99+1)
```

En muchas computadoras para obtener un número entero aleatorio necesitas usar la palabra INT con RND seguido de (1) o (0). A continuación multiplicar por el número de posibilidades que quieres y sumar el primer número. Por ejemplo, todas las instrucciones que se muestran arriba indican a la computadora que escoja un número entre el 1 y el 99.

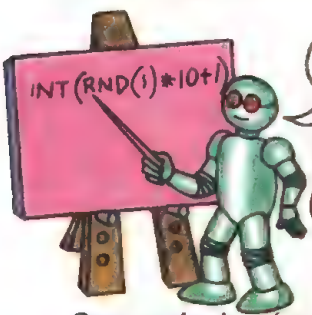



```
PRINT RND(6)+5
PRINT INT(RND(1)*6+5)
PRINT INT(RND(0)*6+5)
PRINT INT(RND*6+5)
```



Estas instrucciones producen un número aleatorio entre el 5 y el 10. Busca el adecuado para tu computadora.

¿Sabrías hacer que tu computadora produjese números aleatorios entre el 10 y el 20?



En este libro escribimos RND de esta manera. Recuerda que debes combiarlo por el que valga a tu computadora.

Problema: alterar un programa

Retrocede para ver el programa para acertar números de la página 13 e intenta modificarlo para que la computadora seleccione un número aleatorio entre el 1 y el 20.

Secuencia de números

```
ACERTAR EL NUMERO
SIGUIENTE DE ESTA
SECUENCIA
  4  13  22
? 31
CORRECTO
```

Intenta escribir un programa para el juego de secuencia de números que se representa aquí. Encima (en el centro) encontrarás una parte del programa.

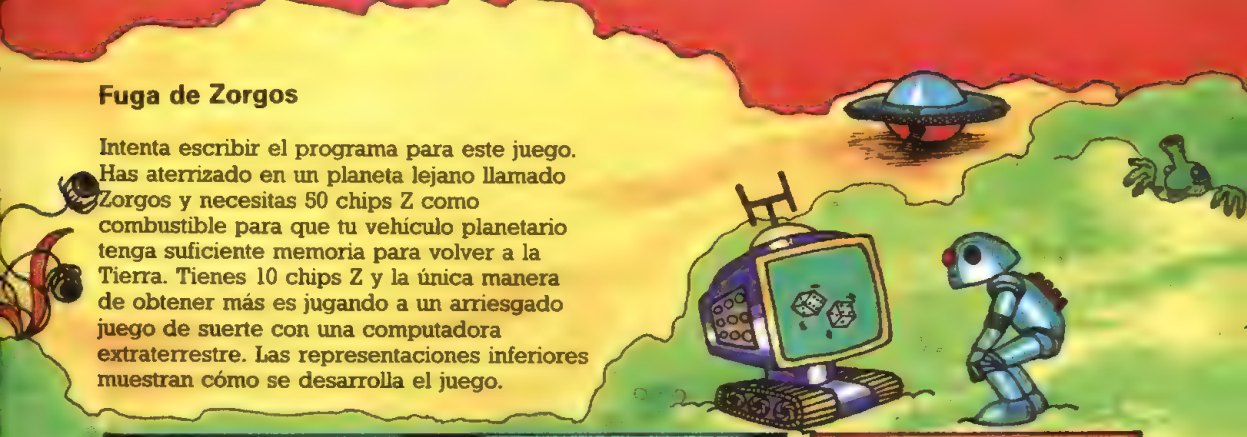
```
* 10 LET X=?
* 20 LET Y=?
  30 FOR I=1 TO 3
  40 PRINT X+I*Y
  50 NEXT I
```

```
PRINT X+I*I
PRINT I*I-Y
PRINT X+Y-I*I
PRINT Y+X^I
```

Tendrás que rellenar las instrucciones de números aleatorios de los pasos 10 y 20 y añadir algunos pasos PRINT e IF/THEN. A la derecha hay algunas sugerencias para cambiar el paso 40 y obtener diferentes secuencias de números. ¿Se te ocurre alguna manera de ampliar el programa para que la computadora tome una secuencia aleatoriamente cada vez que ejecutes el programa?

Fuga de Zorgos

Intenta escribir el programa para este juego. Has aterrizado en un planeta lejano llamado Zorgos y necesitas 50 chips Z como combustible para que tu vehículo planetario tenga suficiente memoria para volver a la Tierra. Tienes 10 chips Z y la única manera de obtener más es jugando a un arriesgado juego de suerte con una computadora extraterrestre. Las representaciones inferiores muestran cómo se desarrolla el juego.



```
TIENES 10 CHIPS
COLOCA TU APUESTA: 5
APRIETA P PARA TIRAR: P
        5    2
CONSERVAS LA APUESTA
```

```
TIENES 10 CHIPS
COLOCA TU APUESTA: 9
APRIETA P PARA TIRAR: P
        6    4
TRIPLICAR LA APUESTA
TIENES 28 CHIPS
```

DADOS	TU APUESTA
DOBLES:	DUPLICAS
10 U 11:	TRIPLICAS
6 O 7:	CONSERVAS LA APUESTA
OTROS:	PIERDES

Apuestas chips Z y dependiendo de los números que salgan cuando la computadora tire dos dados, ganas, pierdes o conservas la apuesta. Si salen dobles, tu apuesta se duplica, y así con las demás posibilidades tal y como se muestra en la tabla de la derecha.

Papel, piedra o tijeras: localiza los errores

Este programa es para jugar a Piedra, Papel o Tijeras con la computadora, con el único inconveniente de que está lleno de errores. Basándote en las anotaciones de la parte derecha que te indican cómo debería ser el programa, intenta localizar los errores y corrígelos para que el programa funcione bien.

```
 10 CLS
 20 LET C=0
 30 LET A=0
 40 LET F=0
 50 LET R=INT(RND(1)*4+1)
 60 IF R=1 THEN LET C$=«PAPEL»
 70 IF R=2 THEN LET C$=«PIEDRA»
 80 IF R=3 C$=«TIJERAS»
 90 PRINT «ESTOY LISTA»
100 PRINT «QUIERES PIEDRA; PAPEL O TIJERAS»
110 INPUT A$
120 PRINT
130 IF C$=«PAPEL» AND A$=«TIJERAS» THEN LET F=1
140 IF C$=«PIEDRA» AND A$=«PAPEL» THEN LET F=1
150 IF A$=«TIJERAS» AND C$=«PIEDRA» THEN LET F=1
160 IF C$=A$ THEN LET F=2
170 PRINT «TU ELIGES»; A$
180 PRINT «YO ELIJO»; C$
190 PRINT «POR TANTO»
200 IF F=0 THEN PRINT «YO GANO»
210 IF F=F THEN PRINT «TU GANAS»
220 IF F=2 THEN PRINT «ES TABLAS»
230 IF F=0 THEN LET A=A+1
240 IF F=1 THEN LET C=C+1
250 PRINT «LA PUNTUACION ES:»
260 PRINT «YO:»; C
270 PRINT «TU:»; A
280 IF C>10 AND A>10 THEN GOTO 40
290 PRINT «ESTE ES EL FIN»
```

C lleva la cuenta de la puntuación de la computadora.
A lleva la cuenta de tu puntuación.
F indica a la computadora quién ha ganado en los pasos 130-160.

La elección que hace la computadora de papel, piedra o tijeras se almacena en C$ y depende del valor de R.

Los pasos IF...THEN calculan quién ha ganado. Si ganas tú, la computadora hace F igual a 1 y si es tablas entonces F igual a 2. Frente a cualquier otra posibilidad F sigue siendo 0, lo que significa que la computadora gana.

La computadora te dice quién gana y calcula la puntuación fijándose en el valor de F.

¿Recuerdas cómo funciona el juego? El papel envuelve a la piedra, la piedra aplasta las tijeras y las tijeras cortan el papel.

Jugar con caracteres

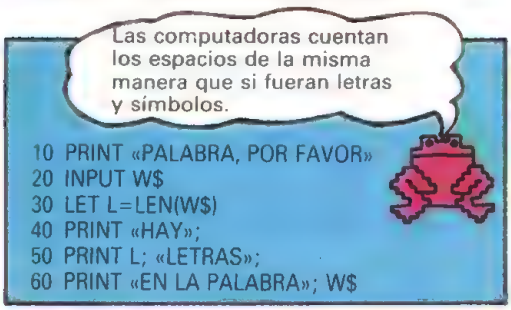
La computadora hace multitud de cosas con los caracteres dentro de la cadena. Los programas de esta página muestran cómo funcionan algunas de las instrucciones que el BASIC usa para manejar cadenas. Si tienes una computadora Sinclair (Timex) tendrás que usar instrucciones diferentes como las que se indican abajo.

```
LET R$=«ROBOT»
PRINT LEFT$(R$,3)
PRINT RIGHT$(R$,3)
LET C$=«ARDILLA»
PRINT LEFT$(C$,4)
PRINT RIGHT$(C$,7)
```

LEFT$ y RIGHT$ sirven para que la computadora tome un número de caracteres de la cadena comenzando por la izquierda o por la derecha. El número de caracteres lo indica poniendo el número entre paréntesis.

```
10 LET K$=«KANGURO»
20 PRINT «POSICION DE»
30 PRINT «PRIMERA LETRA»;
40 INPUT S
50 PRINT «CUANTAS LETRAS»;
60 INPUT N
70 PRINT MID$(K$,S,N)
```

MID$ (medio) sirve para que la computadora tome letras centrales de una cadena. El primer número del paréntesis le indica a la computadora dónde debe comenzar a contar, y el segundo cuántas letras debe poner.



La instrucción LEN cuenta el número de caracteres en una cadena . Ejecuta el programa para ver cómo funciona.

Computadora Sinclair (Timex)

```
LET A$=«ABCDEFGHIJKLMNOP»
PRINT A$(4 TO 6)
DEF
PRINT A$(14 TO 16)
NOP
```

Las computadoras Sinclair (Timex) no usan LEFT$, RIGHT$, ni MID$. En su lugar debes decir a la computadora los tiene que tomar de la manera que indica arriba.

Problema con cadenas

1

```
10 INPUT «PALABRA»; N$
20 LET L=LEN (N$)
30 FOR K=1 TO L
*40 PRINT TAB(K);
   MID$(?,?,?)
50 NEXT K
```

HIPOPOTAMOS

¿Sabrías completar las interrogaciones del paso 40 para hacer que la pantalla aparezca asíú? Pista: Puedes usar MID$ para recoger de uno en uno los caracteres de una cadena.

2

```
RUN
PALABRA, POR FAVOR
?KANGURO
ORUGNAK
```

Intenta hacer que la computadora escriba una palabra al revés utilizando MID$ y un bucle con step-1.

3

```
10 LET S$=
«CIRCULO»
20 LET L=LEN(S$)
30 PRINT S$
40 FOR J=1 TO L
*50 ?
60 NEXT J
```

```
C I R C U L O
I R C U L O C
R C U L O C I
C U L O C I R
U L O C I R C
L O C I R C U
O C I R C U L
```

Utilizando LEFT$ y RIGHT$, intenta completar el paso que falta en este programa para que en la pantalla aparezca algo como esto.

La palabra más larga

En el paso 50 tienes que indicarle a la computadora que almacene la palabra más larga introducida en la variable A$. Intenta usar IF/THEN y LEN

```
10 LET A$=« »
20 PRINT «PALABRAS, POR FAVOR»
30 FOR J=1 TO 5
40 INPUT W$
* 50 ?
60 NEXT J
70 PRINT «PALABRA MAS LARGA:»
80 PRINT A$
```

```
PALABRAS, POR FAVOR
? GATO
? LAGARTIJA
? HIPOPOTAMO
? CABRA
? HORMIGA
PALABRA MAS LARGA:
HIPOPOTAMO
```

Este programa localiza la palabra más larga de una lista de cinco. Completa el paso que falta e intenta ejecutarlo.

Palabra más corta

¿Sabrías escribir un programa para localizar la palabra más corta? Es como el anterior pero necesitas una variable que sea más larga que todas las palabras que vayas a introducir para que la computadora pueda compararlas con ella. Tendrás que cambiar la instrucción IF/THEN.

Puedes llenar la variable con cualquier carácter, como éstos.

```
LET A$=«XXX!!!&&ABC*
**123!!!!XXXXXXXXXX»
```

Editor de palabras

El listado inferior es sobre un programa para editar palabras que te permite introducir frases y luego hacer con ellas todos los cambios que quieras. Antes de ejecutar el programa debes rellenar huecos en los pasos que están marcados con un asterisco. Apóyate en las anotaciones de la derecha del programa.


```
  10 CLS
  20 PRINT «FRASE, POR FAVOR»
  30 INPUT S$
* 40 LET S$=?
  50 PRINT «PALABRA QUE QUIERES CAMBIAR»;
  60 INPUT W$
* 70 LET W$=?
  80 PRINT «PALABRA NUEVA»;
  90 INPUT N$
*100 LET LS=?
*110 LET LW=?
 120 LET A$=« »
 130 LET K=1
 140 IF MID$(S$,?,?)=W$ THEN
 LET A$=S$
*150 IF A$=S$ THEN LET S$=LEFT$
 (?,?)+?+RIGHT$(A$,LS-(K+LW-2))
 160 LET LS=LEN(S$)
 170 LET K=K+1
*180 IF K<=LS-LW+1 THEN GOTO?
 190 PRINT S$
 200 GOTO 50
```

Para entender cómo funciona este programa, escribe una frase en papel y realiza tú mismo las instrucciones del programa.

Haz que la computadora añada un espacio al principio y otro al final de S$ y W$. ¿Entiendes el motivo?

Haz que LS mida lo mismo que la frase (S$) y LW lo mismo que la palabra (W$).

Completa este paso para hacer que la computadora busque la palabra a sustituir (W$) en la frase. Pista: utiliza K para contar los caracteres.

Este paso hace que la computadora calcule el número de caracteres de la izquierda de la palabra que quieres cambiar, luego introduce la nueva palabra y añade el resto de la frase. ¿Sabrías completar las variables que faltan y las cifras?

Más operaciones con caracteres

Dentro de la computadora, los caracteres están representados por números y puedes hacer cosas con los caracteres utilizando estos números de código. La palabra CHR$ convierte un número en un carácter. ASC (o CODE en las computadoras Sinclair (Timex) hace lo contrario convirtiendo un carácter en su número de código. Muchas computadoras utilizan un código standard para los números, denominado código ASCII*, aunque la ZX81 (Timex 1000) tiene su propio código. Encontrarás una tabla de códigos para tu computadora en tu manual.

Utiliza CHR$

```
PRINT CHR$(65)
A
PRINT CHR$(90)
Z
```


```
PRINT CHR$(38)
A
PRINT CHR$(63)
Z
```

Prueba varias sentencias PRINT CHR$ utilizando éstos y otros números. Algunos números son para teclas como SPACE (Espacio) o RETURN (entrar), por lo no aparecerá nada en la pantalla. Puedes averiguar cuáles son mirándolos en tu manual.

Problemas con letras

1

```
*10 FOR K=? TO ?
 20 PRINT CHR$(K);
 30 NEXT K
```

Intenta adivinar los números correctos del bucle para que éste escriba todo el alfabeto.

2

```
abcdefghijklm
nopqrstuvwxyz
```

Intenta escribir un bucle para que tu computadora escriba el abecedario en minúsculas, siempre que las tenga.

3

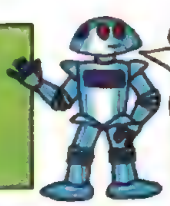
```
IDIPQCHKA
SKQCHUAMSF
SNMQTOPXA
```

¿Sabrías escribir un programa que escribiese una serie de letras aleatorias en la pantalla?

Utiliza ASC o CODE

```
PRINT ASC(«P»)
PRINT ASC(«+»)
PRINT ASC(«  »)
```

```
PRINT CODE(«4»)
PRINT CODE(«U»)
PRINT CODE(«C»)
```



Prueba ASC (o CODE en una computadora Sinclair (Timex) como éstos y otros caracteres para ver qué números produce la computadora.

Compara letras

```
? P,O
O ESTA ANTES QUE P
? L,B
B ESTA ANTES QUE L
? S,C
C ESTA ANTES QUE S
```

Utilizando los símbolos >y <intenta escribir un programa que haga que la computadora compare dos letras y las ordene en orden alfabético.

Transformar en mayúscula

```
 10 PRINT «CUAL ES TU MENSAJE»
 20 INPUT M$
 30 FOR J=1 TO LEN(M$)
 40 LET X$=MID$(M$,J,1)
*50 IF X$>=«a» AND X$<=«z» THEN
 PRINT CHR$(ASC(X$)-?);
*60 IF X$>=«A» AND X$<=«Z» THEN
 PRINT CHR$(ASC(X$)-?);
 70 IF X$<«A» OR X$>«z» THEN PRINT X$;
 80 NEXT J
```

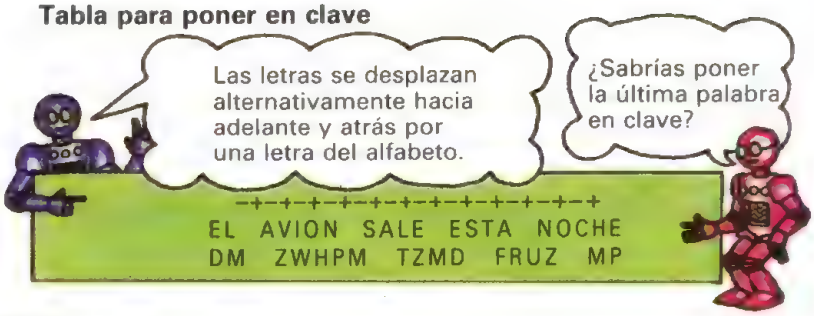
Intenta rellenar las cifras que faltan en los pasos 50 y 60 para hacer que la computadora transforme un mensaje de minúsculas o viceversa. No podrás ejecutar este programa si tu computadora sólo utiliza letras mayúsculas.

* ASCII significa American Standard Code for Information Interchange (Código Estándar Americano para el Intercambio de Información).

Programas para escribir en clave

Estas son algunas ideas para que un programa transforme mensajes poniéndolos en clave. La tabla de la derecha muestra cómo funciona el primer programa. Completa las cifras y símbolos del programa para después ejecutarlo.

Tabla para poner en clave



Programa para escribir en clave

```
  10 PRINT «¿CUAL ES TU MENSAJE?»
  20 INPUT M$
* 30 FOR J=1 TO?
* 40 LET X=?
* 50 IF X<? OR X>? THEN LET N=X:
   GOTO 100
* 60 IF INT(J/2)=J/2 THEN LET N=X? 1
* 70 IF INT(J/2)<>J/2 THEN LET N=X? 1
* 80 IF N<? THEN LET N=N+27
* 90 IF N<? THEN LET N=N-27
*100 PRINT?
 110 NEXT J
```

Haz el bucle del mismo tamaño que el número de caracteres en M$.

Haz que la computadora seleccione letras de una en una y almacene su código numérico ASCII en X.*

Incluye dos números para comprobar cada caracter y estar seguro de que es una letra (los números y los espacios permanecen iguales en esta clave secreta).

Para avanzar las letras suma 1 a X si el contador del bucle (J) es un número par y resta 1 si es impar.

Escribe las letras en clave utilizando CHR$.

Si el número avanzado (N) se sale del alfabeto, mándalo a la otra punta sumando o restando 27.

Número en clave	← N=X+Número de clave → N=N−27
Alfabeto	A B C D E F G H I J K L M N Ñ O P Q R S T U V W X Y Z
Alfabeto en clave	G H I J K L M N Ñ O P Q R S T U V W X Y Z A B C D E F

En esta clave, el alfabeto se traslada un número (N) de letras. Depende de un número de clave. Aquí el número de clave es 6, por lo que el alfabeto se traslada seis letras. Intenta escribir el programa. Puedes utilizar el número de clave que quieras.

Clave de bucle

J=	1	2	3	4	5	6	7	8	9	10	11	12
B	U	S	C	A		A	G	E	N	T	E	X

En esta clave sumas el valor de la variable del contador del bucle (J) al número de código ASCII para cada letra. Intenta escribir dos programas, uno para codificar un mensaje y el otro para decodificarlo.

Clave inversa

BUS CAR AGE NTES ECR ETO

UBC SRAA EGTN EESR CT EO

Para esta clave divide el mensaje en pares de letras e invierte la posición de las letras dentro de cada par incluyendo espacios. Para escribir el programa utiliza un bucle con step 2 y escribe cada par de letras invertido.

*Si tienes un ZX81 (Timex 1000), utiliza los números de código propios del ZX81.

STETE SNUM NEASEJS CEERATE NLCVA ENIEVSR A

Ejercicios con INKEY $

La palabra INKEY$ (letra apartada) hace que la computadora busque en el teclado si se ha apretado alguna tecla, pero al contrario que con INPUT la computadora no te esperará, sino que el programa seguirá ejecutándose. Algunas computadoras utilizan una palabra diferente para INKEY$, por lo que conviene que te asegures antes de realizar los ejercicios de esta página.

Estas son algunas de las palabras utilizadas por otras computadoras en lugar de INKEY$. Comprueba tu manual o la tabla de conversión de la página 47 para ver la instrucción que utiliza tu computadora.

```
10 LET A$=INKEY$
20 IF A$="" THEN PRINT "!";
30 IF A$<>"" THEN PRINT A$;
40 GOTO 10
```

```
HOLA          HOLA          H
OLA       HOLA              HOL
A           HOLA          HOLA
HOLA          HOLA        HO
LA        HOLA              HOLA
```

Intenta ejecutar este programa utilizando la instrucción adecuada para tu computadora. Cuando aprietas una tecla, la computadora escribe ese carácter; si no aprietas ninguna escribirá una interrogación.

Ahora intenta escribir un programa para que tu computadora escriba la palabra HOLA cuando aprietas una tecla y si no aprietas ninguna que vaya dejando espacios libres tal y como se muestra en la representación superior. El programa es similar al de la izquierda.

Haz que la computadora espere

```
10 LET A$=INKEY$(50)
20 IF A$="" THEN
PRINT "  ";
30 IF A$<>"  " THEN
PRINT A$;
40 GOTO 10
```

```
10 LET N=0
20 LET A$=INKEY$
30 IF A$<>"  " THEN GOTO 70
40 LET N=N+1
50 IF N<50 THEN GOTO 20
60 PRINT "  "; : GOTO 10
70 PRINT A$; : GOTO 10
```

¿Sabrías hacer que la computadora espere indefinidamente hasta que aprietes una tecla?

En ocasiones puede ser útil hacer que la computadora espere un poco antes de continuar con el programa. En algunas versiones del BASIC tienes que introducir un número entre paréntesis después de INKEY$ como ves arriba. Este indica en la computadora cuánto debe esperar (en fracciones de segundo) antes de continuar. Si tu computadora no acepta esto puedes hacerlo colocando INKEY$ dentro de un bucle usando GOTO, como se indica a la derecha.

Matemáticas rápidas con chinches (errores)

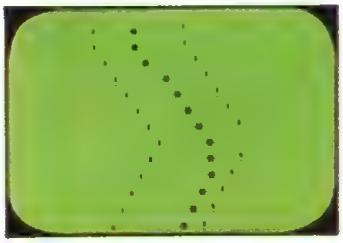
¿Sabrías encontrar las chinches de este programa para corregirlas y que el programa pueda ejecutarse bien? La computadora debe seleccionar dos números aleatorios entre el 1 y el 25. Tú debes sumarlos y apretar cualquier tecla en cuanto la solución correcta aparezca en la pantalla.

```
 10 CLS
 20 PRINT "APRIETA CUALQUIER TECLA CUANDO
 30 PRINT "VEAS LA SOLUCION CORRECTA DE
 LA SUMA"
 40 LET N=0
 50 LET X=INT(RND(1)+25+1)
 60 LET Y=INT(RND(1)* 25+1)
 70 PRINT
 80 PRINT "X;" + ";Y;" = "
 90 LET N=N+1
100 PRINT N
110 LET INKEY$=A$
120 IF A$<>"  " THEN GOTO 180
130 FOR K=1 TO 100; NEXT K
140 IF N<30 THEN GOTO 90
150 PRINT "MALA SUERTE. LA RESPUESTA ES"; X+Y
160 FOR K=1 TO 1000: NEXT T
170 GOTO 30
180 IF N<>X+Y THEN GOTO 150
190 PRINT "SI. LA RESPUESTA ES X+Y"
```

Hay 8 chinches en este programa.

Cómo escribir un juego de coches de choque

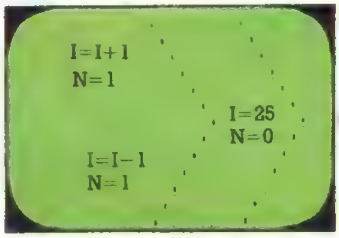
Siguiendo las ideas que te damos, intenta escribir el programa para este juego de coches de choque. La pantalla de la derecha muestra cómo debe ser el juego. Para representar el coche y la carretera, utiliza PRINT TAB con * para el coche y ! para los bordes de la carretera. La carretera debe ir en zig-zag a lo largo de la pantalla mientras tú conduces el coche con dos teclas para impedir que choque con los bordes de la carretera.

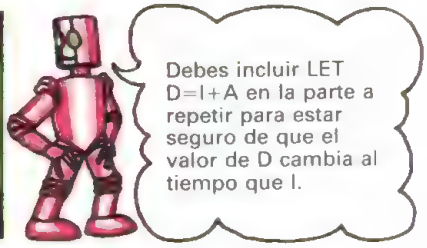


1. Establecer las variables

Necesitas cuatro variables C, I, A y D para calcular la posición TAB del coche y de los lados de la carretera. Quizá tendrás que cambiar los números que se dan arriba para que encaje en tu pantalla. Borra la pantalla y crea estas variables en los primeros cinco pasos.

2. Representación de la carretera

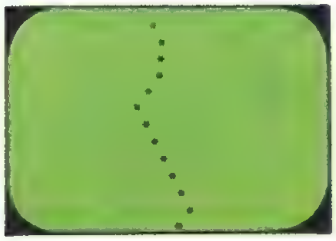




Ahora haz que la computadora represente la carretera y el coche utilizando PRINT TAB con una variable y un símbolo. Utiliza GOTO para repetir la instrucción y dibujar una carretera larga y recta.

Para hacer que la carretera tenga curvas debes incluir algunas para cambiar el valor de I cada vez que se repiten las instrucciones PRINT TAB. También debes asegurarte de que la carretera no se sale de la pantalla. Para esto necesitarás otra variable (N). Esta deberá ser 1 si I<=1 y 0 si I>=la anchura de tu pantalla. Luego indícale a la computadora que sume o reste 1 de I dependiendo del valor de N.

3. Conducir el coche



Para conducir el coche necesitas un paso INKEY$. También debes escoger dos teclas (tales como < y >) y sumar o restar 1 de C según qué tecla se apriete. Si el programa va demasiado deprisa para poder controlar el coche, añade un bucle de retardamiento.

Finalmente debes comprobar si el coche ha chocado con los bordes de la carretera comparando C con I y con D. Si hay un choque debes decírselo al jugador y volver a empezar.

Problemas de DATA

Uno de los métodos más difíciles de dar grandes cantidades de información a la computadora es mediante las palabras READ (leer) y DATA (datos). Un paso DATA contiene una lista de palabras o números que se almacenan en una o más variables según lo indique la instrucción READ. Si tienes un ZX81 (Timex 1000) no podrás ejecutar los programas de estas dos páginas, ya que tu computadora no utiliza estas instrucciones, aunque en la página 24 te ofrecemos un sistema para que puedas almacenar la información de un DATA utilizando una matriz.

Leer el DATA

Separa cada dato con una coma.

```
10 FOR I=1 TO 7
20 READ X,X$
30 PRINT X; «  »; X$
40 NEXT I
50 DATA 13, PECES, 77, RANAS, 91, CARACOLES
60 DATA 23, GATOS, 62, PERROS, 2, RATONES, 1, RATA
```

Algunas computadoras exigirán que estén los datos entre comillas.

Ejecuta este programa para ver cómo funciona READ y DATA. La palabra READ va seguida de dos variables y cada vez que se repite el bucle la computadora almacena el siguiente par de datgs en las variables X y X$.

Comprobación de nombres

Sustituye la interrogación del paso 50 con el último nombre de tu lista.

```
10 PRINT «NOMBRE, POR FAVORW;
20 INPUT N$
30 READ X$
40 IF X$=N$ THEN GOTO 70
* 50 IF X$=«?» THEN PRINT «TU
   NOMBRE NO ESTA EN LA LISTA»: STOP
60 GOTO 30
70 PRINT «OK TU NOMBRE ESTA EN
   LA LISTA»
* 80 DATA?
```

En este programa, la computadora te pide tu nombre para ver si se halla en una lista de nombres almacenada como data. Pon todos los datos que quieras en el paso 80, pero recuerda que en paso el 50 debes poner el último nombre de la lista para decirle a la computadora que pare el programa después de leerlo.

Volver DATA al principio: RESTORE

```
* 10 FOR J=? TO?
* 20 FOR I=? TO ?
* 30 READ N$
  40 IF LEFT$(N$,1)=CHR$(?) THEN PRINT N$
  50 NEXT I
  60 RESTORE
  70 NEXT J
  80 DATA VERA, XAVIER, ZACARIAS, HORACIO, BIGGLER, BILL, BEN
  90 DATA TOPSY, TIM, POPEYE, JIM, HARRY, GEORGE, DELILAH
 100 DATA LOVEDAY, HONORA, SAMPSON, SAUL, TABITHA
```

Haz que el bucle J mida igual que el alfabeto utilizando el número de código del carácter correspondiente.

Haz que la longitud del bucle I sea el número de datos.

¿Sabrías completar este programa para que ordenase los nombres alfabéticamente? Los nombres están almacenados en pasos DATA; RESTORE (volver a empezar) hace que la computadora vuelva al principio de la lista de datos cada vez que se repite el bucle J. Intenta completar las variables y números que faltan para poder ejecutar el programa.

Localiza el chinche

1

```
10 FOR K=1 TO 6
20 READ N: PRINT N
30 NEXT K
40 DATA 01-232,22-36-41,341-2241/2
50 DATA 97-24-11,47-29-01,236-4013
```

En este programa los pasos data contienen números de teléfono. ¿Puedes descubrir los chinches del programa? Si es así, corrígelos.

2

```
10 READ X
20 PRINT X
30 GOTO 10
40 DATA 1,461,89266,1471,4462,1,3
50 DATA 53,80,241,90,371,825,33,13
```

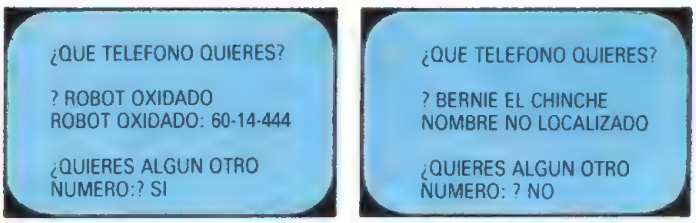
Si no encuentras el chinche de este programa, prueba a ejecutarlo. La computadora escribirá un mensaje de error diciendo lo que está mal. ¿Se te ocurre una manera fácil de resolver este programa?

El café de Joe

En la parte superior izquierda está el menú del café de Joe. Utilizando los precios y los datos, intenta hacer un programa como el de la pantalla de la derecha que te diga lo que puedes tomar con una determinada cantidad de dinero. Puedes añadir más platos al menú.

Agenda de teléfonos

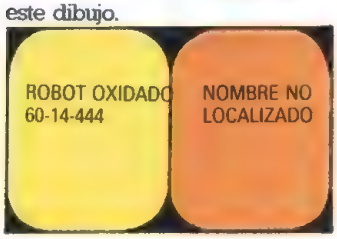
Debajo hay instrucciones para escribir un programa que sirva de agenda telefónica. Las pantallas de la derecha te muestran cómo funciona. Intenta escribir el programa.



1. Haz una lista con los nombres de tus amigos y sus números de teléfono e introdúcelos en una instrucción DATA como se muestra en este dibujo.

2. Utiliza PRINT par hacer que la computadora te pregunte qué número quieres e INPUT para respuesta.

3. Utilizar READ para buscar el nombre dentro de un bucle. Utiliza variables separadas para nombres y números.



4. Escribe el nombre y el número (que debe ser el siguiente dato de la sentencia data) o bien indica que el nombre no está en la lista.

5. Haz que la computadora te pregunte si quieres otros números. Utiliza INPUT para tu respuesta.

6. Según la respuesta: RESTORE vuelve al principio el dato o bien se da por terminado el programa.

Uso de matrices

Una forma cómoda de almacenar datos es con matrices. Puedes imaginarte una matriz como un conjunto de variables donde cada dato se almacena en un compartimento numerado. Los datos se denominan elementos. Puedes referirte a un elemento utilizando el nombre de la matriz y su número de compartimento también llamado subíndice o celda.

Matriz numérica

Esta es una matriz numérica llamada N. Contiene seis elementos. Debes indicarle a la computadora qué tamaño va a tener la matriz para que pueda reservar la memoria necesaria. Para hacer esto se utiliza la sentencia DIM seguida del nombre de la matriz y el número de elementos que contiene. Esto se denomina dimensionar una matriz.

```
* 10 DIM?
  20 FOR K=1 TO 6
  30 READ N(K)
  40 NEXT K
* 50 DATA?
```

Para ZX81 (Timex 1000)

```
* 10 DIM?
  20 FOR K=1 TO 6
  30 INPUT N(K)
  40 NEXT K
```

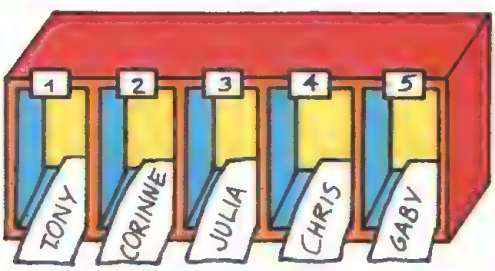
Para introducir datos en una matriz puedes utilizar un bucle con READ/DATA. Intenta completar el programa de la izquierda para que almacene toda la información del dibujo superior en una matriz. Si tienes un ZX81 (Timex 1000), utiliza el programa de la derecha que utiliza una serie de introducciones INPUT para llenar la matriz.

```
N(1) IS 1066
N(2) IS 1216
N(3) IS 1485
N(4) IS 1603
N(5) IS 1665
N(6) IS 1959
```

```
1485  1216
1959  1485
1939  1665
1603  1603
1485  1216
```

Ahora intenta escribir un programa que ponga en la pantalla todos lo datos almacenados en la matriz. Utiliza PRINT y una variable de bucle que equivalga al número de compartimentos de la matriz. En la pantalla de la derecha, la computadora escribe elementos de una matriz de forma aleatoria utilizando un número aleatorio que busca como número de compartimento.

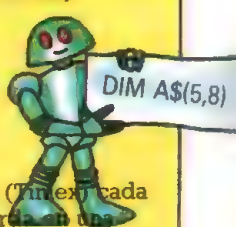
Matriz de caracteres



Esta es una matriz de caracteres (N$) Contiene cinco nombres, es decir, cinco elementos. A menos que tengas una computadora Sinclair (Timex), has de usar las matrices de caracteres de igual forma que las numéricas. Escribe un programa simple para almacenar los datos en esta matriz y que a continuación muestre los datos en la pantalla.

Computadora Sinclair (Timex)

```
ARRAY A$
1 GATO
2 PEZ
3 ELEFANTE
4 RATA
5 BURRO
```



En las computadoras Sinclair (Timex) cada cadena de caracteres se guarda en una línea y cada elemento de la línea contiene un carácter. Para dimensionar una matriz tienes que decirle a la computadora cuántas cadenas (líneas) hay y cuántos elementos. Todas las líneas han de tener la misma longitud, por lo que conviene contar los caracteres de la cadena más larga y utilizar este número en la sentencia DIM. La computadora hará todas las demás cadenas de la misma longitud añadiendo espacios en blanco.

Calculador de meses

```
* 10? ]
  20 FOR K=1 TO 12
```
Dimensiona la matriz.

```
* 30? ]
  40 NEXT K
  50 PRINT «NUMERO DE MES»;
  60 INPUT N
* 70 PRINT M$(?); «TIENE»;
* 80 PRINT D(?); «DIAS»
```
READ para los datos de dos matrices M$ y D.

Completa los números para que busque los datos correctos en las matrices.

```
* 90 DATA?
*100 DATA?
*110 DATA?
```
En estas instrucciones data pon el nombre de cada mes seguido del número de días que tiene.

¿Sabrías escribir este programa para que cuando escribas el número de un mes la computadora escriba su nombre y el número de días que tiene? Al lado del programa hay algunas pistas que te pueden ayudar.

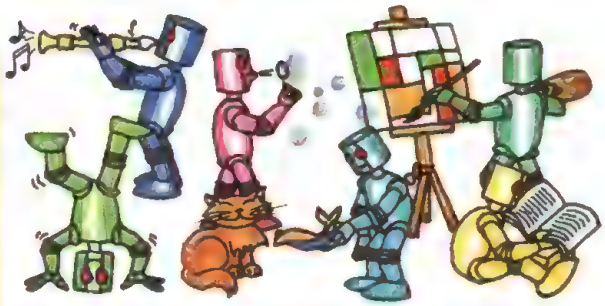
```
¿CUAL ES EL NOMBRE DEL
MES? JULIO

JULIO TIENE 31 DIAS
```

Ahora altera este programa para que la computadora te pregunte el nombre de un mes y a continuación te diga el número de días que tiene. Utiliza en bucle IF/THEN para buscar en M$ el nombre del mes. Utiliza el número de compartimento como variable del bucle para tomar el dato correcto almacenado en D.

Programa para decidir qué haces

```
ESCOGE UN NUMERO DEL 1 AL 10
? 9
POR QUE NO HACES EL PINO
OK? NO

ESCOGE UN NUMERO DEL 1 AL 10
? 5
POR QUE NO DAS COMIDA AL GATO
OK? SI
```

Esta es una idea para hacer un programa que te puede ser útil cuando no se te ocurra qué hacer. Para ejecutar el programa, elige un número y, según el que elijas, la computadora te dará diferentes sugerencias.



Para escribir el programa necesitas una matriz de caracteres (I$) llena con diez ideas. INPUT para introducir un número en la variable (N) y haz que la computadora use una de las ideas utilizando N como número de compartimento de I$.

Veinte preguntas

```
¿CUANTAS PATAS TIENE UN
CIEMPIES? 100
CORRECTO

¿CUANTOS JUGADORES HAY EN
UN EQUIPO DE FUTBOL?
? 10
NO TONTO
LA RESPUESTA ES 11
```

Esta pantalla muestra un juego de preguntas. Para escribir el programa piensa en 20 preguntas e introdúcelas en la matriz de caracteres. Pon las respuestas en otra matriz. Los números de compartimento de las dos matrices deben unir preguntas con respuestas.

Tabla de números aleatorios

```
 1  **********
 2  ***************
 3  **********
 4  **************
 5  **********
 6  ********
 7  ***********
 8  **********
 9  ************
10  ***********
```

```
10 LET N=0
20 DIM A(10)
30 FOR K=1 TO 10
40 LET A(K)=0
50 NEXT K
60 LET R=INT(RND(1)*10+1)
70 LET A(R)=A(R)+1
80 LET N=N+1
90 IF N<100 THEN GOTO 60
```

La computadora da a cada elemento de la matriz a un valor inicial 0.

Este programa usa una matriz para almacenar dos datos de una tabla. La computadora elige aleatoriamente 100 números del 1 al 10. Cada elemento de la matriz (A) lleva la cuenta del número de veces que se escoge cada número. Intenta añadir al programa unos pasos para que la computadora represente una tabla como la que se muestra arriba que señale con una estrella cada número elegido. Para hacer esto, haz un bucle que se repita 10 veces con otro bucle en su interior. El bucle anidado deberá representar una línea de la tabla cada vez.

Escribir subrutinas

Una subrutina es una parte del programa que desempeña una determinada labor que ha de ejecutarse varias veces a lo largo del programa. La palabra GOSUB (ir de subrutina) seguida del número del primer paso de la subrutina le dice a la computadora que debe saltar a esa subrutina. La computadora ejecuta la subrutina hasta que encuentra la palabra RETURN (volver). Esta instrucción la manda el programa principal justo al paso posterior al que contenía la instrucción GOSUB.

Intenta ejecutar este programa para entender cómo funcionan los comandos GOSUB/RETURN.

```
10 PRINT «HOLA»
20 GOSUB 1010
30 PRINT «HOLA DE NUEVO»;
40 GOSUB 1010
50 PRINT «Y QUE ES ESTO?»
60 GOSUB 2020
70 STOP
1010 PRINT «ESTO ES UNA SUBRUTINA»
1020 RETURN
2020 PRINT «OTRA SUBRUTINA»
2030 RETURN
```

Encuesta sobre helados	
MELON	16
PLATANO	11
JENJIBRE	8
PEPINILLOS	1
CHICLE	18

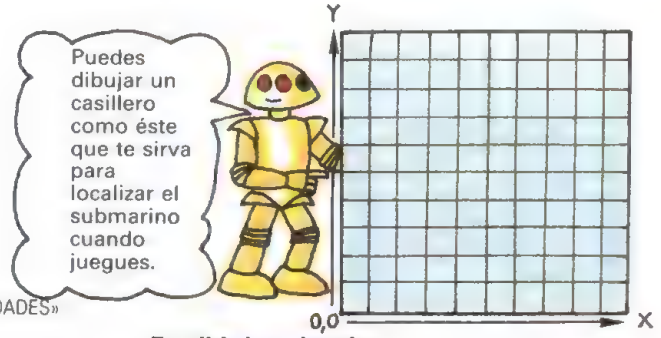
```
MELON          ****************
PLATANO        **********
JENJIBRE       ********
PEPINILLOS     *
CHICLE         ******************
```

La tabla superior izquierda es el resultado de una encuesta para averiguar el sabor más popular de entre seis nuevos gustos. Escribe un programa para representar la información como en la pantalla de la derecha. Utiliza un bucle para leer los datos con dos matrices y dentro de él pon una instrucción que mande a la computadora a una subrutina que represente una a una cada fila de la tabla.

Hunde el submarino

El programa que viene a continuación es para un juego llamado «Hunde el submarino». La computadora decide su posición eligiendo dos números aleatorios que serán las coordenadas X e Y. Tienes cuatro intentos para hundir el submarino adivinando las coordenadas X e Y. Si tu intento es fallido, la computadora salta a una subrutina que dice en qué dirección debes apuntar. Antes de ejecutar el programa debes escribir la subrutina.

```
10 CLS
20 LET N=0
30 LET X=INT(RND(1)*10+1)
40 LET Y=INT(RND(1)*10+1)
50 LET N=N+1

60 PRINT «INTENTO»; N; «  »;
70 INPUT A,B
80 IF A=X AND B=Y THEN GOTO 170

90 GOSUB 200
100 PRINT

110 IF N<=4 THEN GOTO 50
120 PRINT «NO TE QUEDAN MAS OPORTUNIDADES»

130 PRINT «QUIERES VOLVER A JUGAR»

140 INPUT R$
150 IF LEFT$(R$,1)=«Y» THEN GOTO 10
160 STOP

170 PRINT «DISTE AL SUBMARINO EN»; N; «INTENTOS»

180 GOTO 130
```



Escribir la subrutina

Necesitas varios pasos IF/THEN para comparar los números (A, B) con la posición del submarino (X, Y) y que aparezcan en la pantalla los mensajes correspondientes. Si B es menor que Y apunta al norte; si A es menor que X apunta al Este, y así con las demás posibilidades.

Ejecución del programa

```
INTENTO 1 ?3,2
FALLASTE
DISPARA AL NORESTE
```

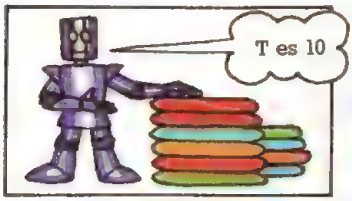
```
INTENTO 2 ?3,3
FALLASTE
DISPARA AL ESTE
```

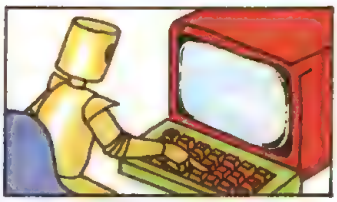
```
INTENTO 3 ?5,3
FALLASTE
DISPARA AL OESTE
```

Programa para una máquina tragaperrras

¿Sabrías escribir un programa para que tu computadora funcione como una máquina tragaperras? La pantalla de la derecha muestra cómo aparecería el juego. Debes comenzar con 10 fichas y te cuesta una ficha cada vez que juegas. Aprieta cualquier tecla y la computadora escribe el nombre de tres «frutos» elegidos aleatoriamente. Intenta seguir los consejos que te damos para poder escribir el programa.

```
TIENES 10 FICHAS
APRIETA CUALQUIER TECLA
PARA JUGAR: P
CAMPANA  CAMPANA  LIMON

2 CAMPANAS
GANAS 2 FICHAS
AHORA TIENES 11 FICHAS
APRIETA CUALQUIER TECLA
PARA JUGAR: B
```

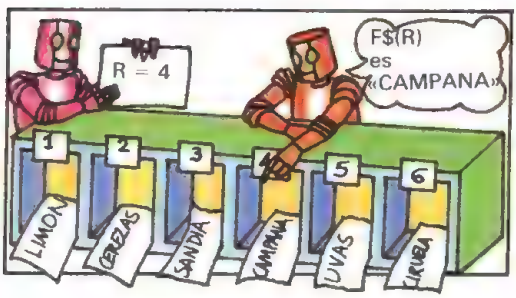




1. En primer lugar, dimensiona la matriz FH para que contenga 6 nombres de «frutas» (limón, cereza, ciruela, sandía, uvas y campana). Almacena estos datos en F$ usando READ/DATA (o INPUT).

2. Borra la pantalla e introduce una variable T para que lleve la cuenta del número de fichas. Dale un valor inicial de 10 e indícale al jugador cuántas tiene.

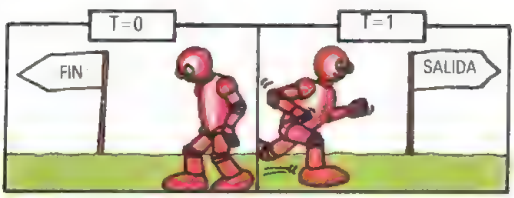
3. Utiliza INKEY$ para hacer que la computadora espere a que el jugador apriete cualquier tecla para empezar. Resta 1 de T (cuesta una ficha jugar) y vuelve a borrar la pantalla.



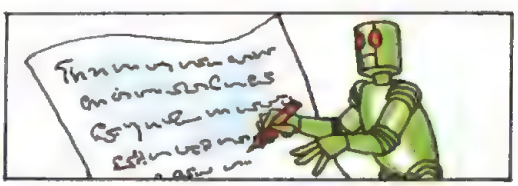
3 CEREZAS:	ESPECIAL
3 DE CUALQUIER CLASE	GANAS 5 FICHAS
2 DE CUALQUIER CLASE	GANAS 2 FICHAS

4. Haz que la computadora elija un número aleatorio entre el uno y el seis y lo almacene en R. Utiliza R para como el número de compartimento de F$ para seleccionar una fruta y almacenarla en una variable A$. Repite este proceso dos veces más para seleccionar dos nuevas frutas y almacénalas en B$ y C$.

5. A continuación deben aparecer los nombres de las frutas en la pantalla y que la computadora calcule si ha ganado algo o no. Como se muestra arriba, hay tres formas de ganar. Para cada una debes mandar a la computadora a una subrutina diferente para que busque el mensaje correcto y ajuste el valor de T.



6. Comprueba que el jugador tiene al menos una ficha. Si no le queda ninguna (T = 0), el programa debería pararse. Si le queda alguna, haz que vuelva a jugar.

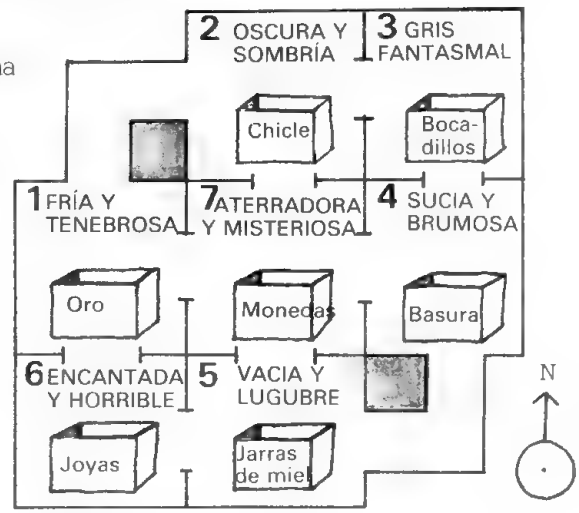


7. Si usas READ al principio del programa no te olvides de escribir los pasos DATA. Esto es preferible escribirlos al final del programa.

Cómo escribir un programa sobre la caza del tesoro

En las páginas siguientes tienes las instrucciones necesarias para escribir paso a paso un juego sobre la caza del tesoro. El programa es algo complicado, por lo que sigue atentamente las sugerencias y comprueba cada apartado según los termines. Si te quedas atascado en alguna parte del programa puedes mirar la solución y continuar con el resto del programa.

Antes de comenzar a escribir el programa debes saber cómo jugar, así es que lee esta sección antes de comenzar.

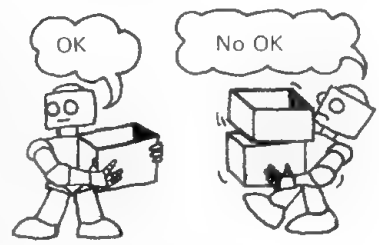


Cómo jugar

El mapa de la derecha muestra un plano de siete habitaciones. Cada una tiene una descripción diferente y una caja llena de tesoros. La información de este mapa queda almacenada en la memoria de la computadora pero el mapa no aparece en la pantalla. Esto significa que no sabrás la posición de cada habitación elegida al azar, el objetivo del juego es reunir todo el tesoro en una habitación en un número limitado de movidas.

```
ESTAS SON LAS PALABRAS QUE LA COMPUTADORA ENTIENDE

N,E,S,O    : MUEVE HACIA NORTE/ESTE/SUR U OESTE
COGER      : RECOGER UN TESORO
DEJAR      : DEJAR UN TESORO
POSICION   : INDICA LA SITUACION DEL TESORO EN ESE MOMENTO
AYUDA      : TE INDICA LO QUE DEBES HACER
```

La pantalla superior muestra las palabras que puedes usar para jugar y lo que significan. Sólo puedes introducir una instrucción en cada jugada y no puedes permitirte el cometer errores.



Para que sea más complicado, sólo puedes llevar las cajas de tesoros de una en una.

Ejemplo de programa

1

```
ESTAS EN LA HABITACION 4
ES SUCIA Y BRUMOSA
CONTIENE BASURA

¿QUE QUIERES HACER?
? RECOGER

OK LLEVAS LA BASURA
```

2

```
SIGUES EN LA HABITACION 4
¿QUE QUIERES HACER?
? O

OK
```

3

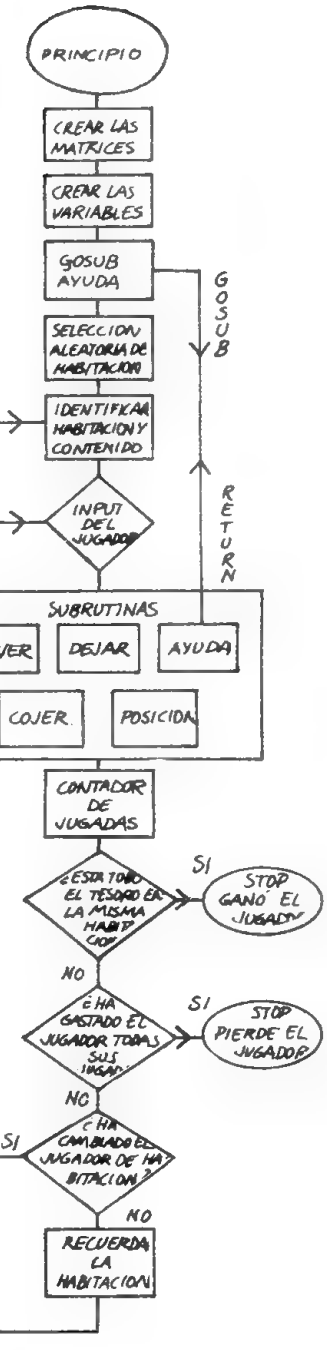
```
ESTAS EN LA HABITACION 7
ES ATERRADORA Y MISTERIOSA
CONTIENE MONEDAS

¿QUE QUIERES HACER?
? DEJAR

OK LA BASURA COLOCADA EN LA HABITACION
```

4

```
SIGUES EN LA HABITACION 7
¿QUE QUIERES HACER?
? S

OK
```

Escribir el programa

El diagrama de la derecha es un diagrama de flujo. Muestra la estructura del programa con cada paso importante en la lista que va de arriba a abajo. Las instrucciones de la parte inferior de las siguientes páginas te guiarán en cada etapa. No te preocupes si te dicen que saltes a diferentes partes del programa fuera de la secuencia. Si sigues la numeración de paso sugerida puedes estar seguro de que acabarás el programa en el orden correcto.



1 Crea las matrices y lee los datos (Pasos 100-250)

Tienes que introducir toda la información del mapa de la página anterior en la memoria de la computadora.

Matrices N, E, S, O

Primero necesitas cuatro matrices llamadas N, E, S y O. Los datos en ellas indican a la computadora qué habitación hay al norte, al este, al sur y al oeste de cada habitación. Dimensiona estas matrices en el primer paso (100)*. Cada una tiene siete elementos.

	N	E	S	O
1	2	7	6	0
2	0	3	7	1
3				
4				
5				
6				
7				

El cero significa que no hay ninguna habitación en esa dirección. Por ejemplo, no la hay al oeste de la habitación 1 o al norte de la habitación 2.

Haz una tabla como esta para calcular los datos. Los números de la primera columna son los «números de compartimento» de las matrices y representan al número de habitaciones. Para rellenar cada casilla de la tabla, mira en el mapa el número de la habitación que hay al norte, el este, al sur y al oeste de las habitaciones listados en el lado de la tabla. Las dos primeras filas ya han sido hechas.

Añade una instrucción de borrado de pantalla

Lee los datos

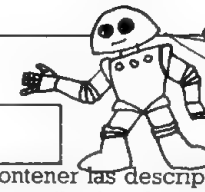
```
110 FOR K=1 TO 7
120 READ N(K),E(K),S(K),O(K)
130 NEXT K

2000 DATA 2,7,6,0
```

Si tienes un ZX81 (Timex 1000) necesitarás usar además una instrucción INPUT para cada matriz.

A continuación necesitas un bucle como el superior para leer los datos. Utiliza la tabla para introducir los datos en siete pasos DATA. El primero se muestra arriba.

*Si tienes un ZX81 (Timex 1000), sólo puedes dimensionar una matriz por paso, así es que comienza en el paso 10.

Matriz D$

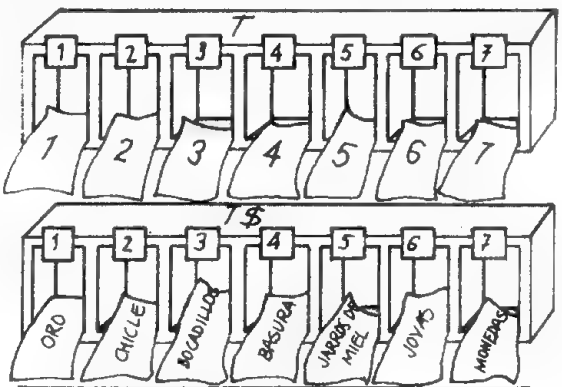


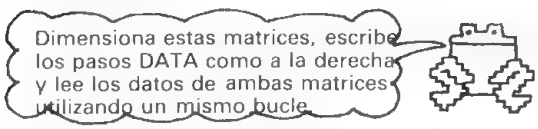

```
2100 DATA FRIA Y TENEBROSA, OSCURA Y SOMBRIA
```

A continuación necesitarás una matriz llamada D$ para contener las descripciones de las habitaciones. Dimensiónala al principio del programa y escribe los datos tal y como se muestra arriba, usando las descripciones del mapa.

Matrices T$ y T

Ahora necesitas dos nuevas matrices. T$ para contener los nombres de los tesoros y T para saber dónde está cada tesoro. El número almacenado en cada compartimento de T es de la habitación donde se almacena el tesoro de T$ con el mismo número de compartimento. Por ejemplo, al principio del juego T(2) es 2, siendo ésta la habitación donde el tesoro T$(2) está almacenado (el chicle). Durante el juego, los números de T cambian según sean trasladadas las cajas de tesoros.






```
2200 DATA ORO,1,CHICLE,2
2210 DATA BOCADILLOS,3,BASURA,4
```

Variables

M	C	F	WXY
Número de jugadas que lleva el jugador.	Indica si el jugador lleva algo.	Una variable «flag» (bandera o indicador). Más tarde verás cómo funciona.	Variables para contener datos temporales.

Las cajas superiores dan los nombres de las variables que necesitarás más adelante. Comenzando en el paso 300, asigna a cada una de estas variables el valor inicial 0.

2. Subrutina Ayuda (pasos 1000-1120)

```
HAY 7 HABITACIONES CON UN
TESORO EN CADA UNA. DEBES
REUNIR TODOS LOS TESOROS EN
UNA SOLA HABITACION.
```

A continuación necesitarás una subrutina que explique las reglas del juego. Pon una instrucción GOSUB en el paso 300 y comienza la subrutina en el paso 1000. Escribe las instrucciones que se muestran en la pantalla (izquierda) y las instrucciones que entiende la computadora (pág. 28).

3. Selección aleatoria de una habitación (paso 350)

El número de la habitación en la que se encuentra el jugador, se almacena en una variable R. Para escoger de qué habitación debe partir, haz que la computadora elija aleatoriamente un número entre el 1 y el 7 y lo almacene en R en el paso 350.

4. Identificar la habitación y su contenido (pasos 400-470)

```
ESTAS EN LA HABITACION 7
ES ATERRADORA Y MISTERIOSA
CONTIENE: MONEDAS
```

```
ESTAS EN LA HABITACION 4
ES SUCIA Y BRUMOSA
CONTIENE: NADA
```

```
ESTAS EN LA HABITACION 1
ES FRIA Y TENEBROSA
CONTIENE: ORO
```

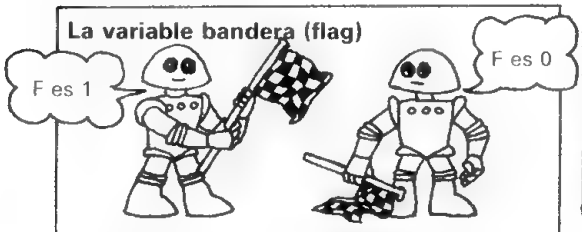
Para empezar, dile al jugador en qué habitación se halla. Luego describe la habitación usando R como «número de compartimento» de D$. Las pantallas superiores muestran cómo debería aparecer la tuya.

Identificación del contenido de la habitación

```
430 FOR K  1 TO 7
440 IF T(K)  R THEN PRINT T$(K):
LET F  1
450 NEXT K
460 IF F  0 THEN PRINT «NADA»
470 LET F  0
```

Para saber que hay en la habitación R tienes que buscar en la matriz T que tiene almacenada el número de habitación en que se halla cada tesoro. Para hacer esto puedes usar un bucle como se muestra arriba. El paso IF/THEN busca para ver si algún número almacenado en T coincide con el número R. Si coinciden la computadora, escribe el nombre de la caja del tesoro y asigna el valor 1 a la variable F.

La variable bandera (flag)



La variable F indica a la computadora si hay algún tesoro en la habitación una vez que ésta ha sido sacada por el bucle. Funciona como una bandera. Cuando el paso IF/THEN es verdadero, la bandera sube (es decir, F es 1), mientras que si es falso F, continúa siendo 0. Recuerda volverla a 0 en el paso 470, ya que volverás a necesitarla.

5. Entrada (Input) del jugador (pasos 500-560)

A continuación pregunta al jugador qué quiere hacer e introduce un paso INPUT que permita que el jugador escriba una instrucción (cómo TOMAR o DEJAR) que se almacene en A$. Analiza lo que escribe el jugador con cinco pasos GOSUB que manden a la computadora a diferentes subrutinas. Pon los números de paso cuando hayas calculado dónde deben ir las subrutinas en el programa.

6. Subrutina mover

(pasos 1200-1260)

Intenta escribir la subrutina mover basándote en este diagrama de flujo

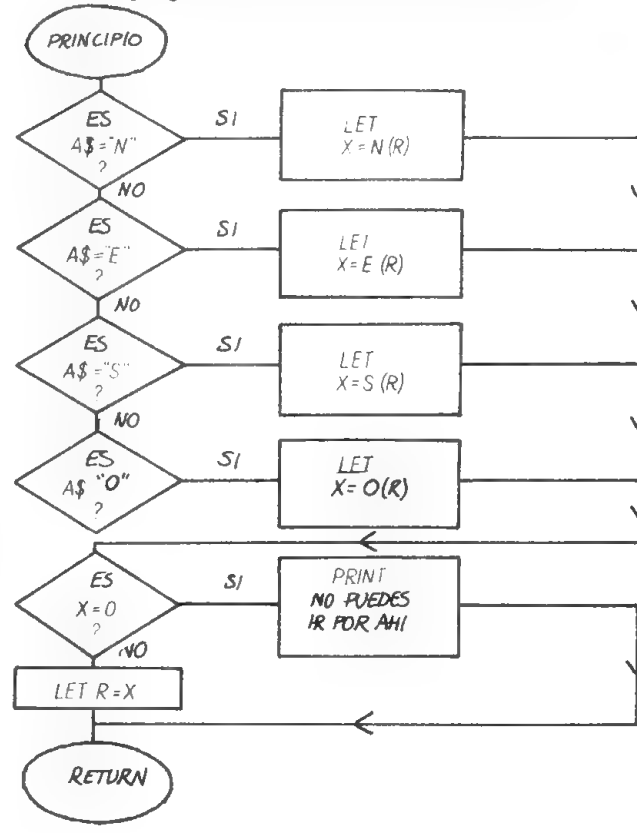
1. Para trasladarse el jugador escribe N, E, S u O; para saber cuál de ellos ha escrito utiliza cuatro pasos IF/THEN.

2. Para encontrar el nuevo número de habitación utiliza R como «número de compartimento» de la matriz correspondiente (N, E, S u O) y almacénalo en una variable temporal X.

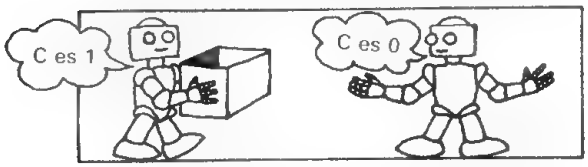
3. Después tienes que comprobar X para estar seguro de que el jugador puede trasladarse en esa dirección. Si X es O quiere decir que no hay ninguna habitación en esa dirección, por lo que tendrás que escribir un mensaje que lo diga y usar un GOTO para mandar a la computadora a una instrucción RETURN al final de la subrutina.

4. Si X no es 0 debes poner el número de la nueva habitación en R haciendo que R sea igual a X.

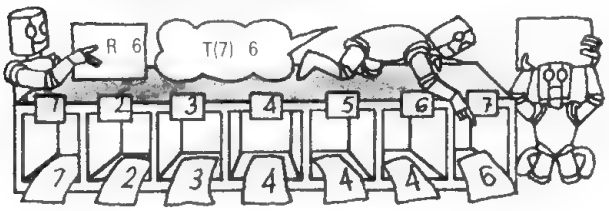
5. Pon RETURN al final de la subrutina.



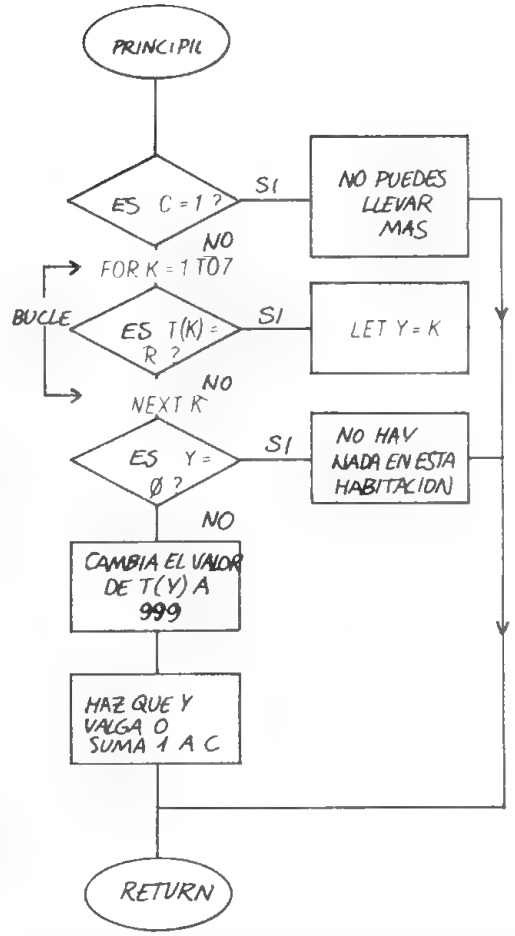
7. Subrutina tomar (pasos 1300-1380)



1. El jugador sólo puede llevar las cajas de tesoros de una en una, así es que primero comprueba si el jugador lleva algo. Para hacerlo, comprueba el valor de C que es 1 cuando el jugador ha tomado algún tesoro. Si C es 1, escribe un mensaje y manda a la computadora a RETURN.



2. Para comprobar si hay algún tesoro en la habitación, escribe un bucle utilizando IF/THEN para buscar en la matriz T. Si el número en T equivale al número de habitación (R), debes almacenar el «número de compartimento» de T (representado por K) en una variable temporal llamada Y. Según la computadora vaya ejecutando el bucle localizará varios números iguales a R. En este caso Y debe contener el «número de compartimento» del número de estos números T cuando la computadora haya terminado el bucle.



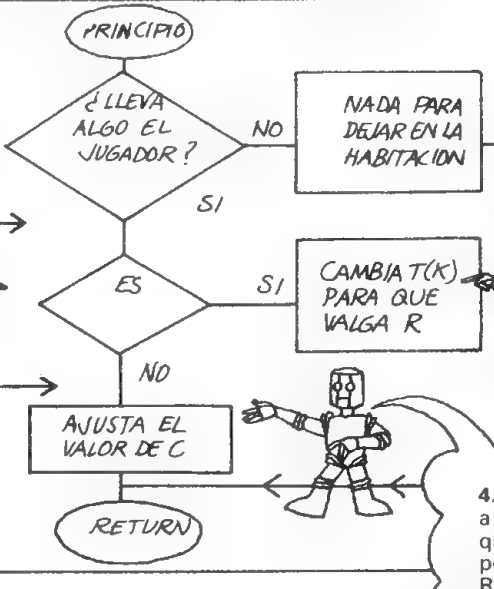
3. Si Y es 0 no hay ningún tesoro en la habitación, así es que debes mandar a la computadora a RETURN. En cualquier otro caso el «número de compartimento» almacenado en Y indica a la computadora qué tesoro tomar. Cambiar el valor T(Y) a 999, un número que indica qué se está llevando. Asigna el valor 1 a C, vuelve Y al valor 0 y pon RETURN al final de la subrutina.

8. Subrutina Dejar (pasos 1400-1460)

1. Primero comprueba si el jugador lleva algo revisando el valor de C.

BUCLE

2. Averigua qué caja de tesoro lleva el jugador buscando 999 en la matriz T.



3. Si T(K) es 999, cambia su valor para que sea igual al número de habitación (R) y vuelve a hacer que C sea 0 para indicar que el jugador no lleva nada.

4. Añade pasos PRINT a la rutina donde creas que son necesarios y pon una instrucción RETURN al final.

9. Subrutina posición (pasos 1500-1590)

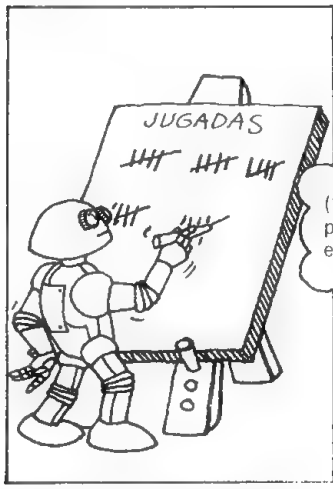
Esta rutina informa al jugador de la posición en ese momento de cada caja de tesoro. Intenta escribir el programa basándote en la muestra de lo que debería salir que hay en las dos pantallas inferiores. Primero traza un diagrama de flujo.

```
LLEVAS: ORO
CONTENIDO DE LAS HABITACIONES:
2: CHICLES
3: BOCADILLOS
4: BASURA
5: JARROS DE MIEL
6: JOYAS
7: MONEDAS
```

```
LLEVAS: NADA
CONTENIDO DE LAS HABITACIONES
7: ORO
7: CHICLE
7: BOCADILLOS
7: BASURA
7: JARROS DE MIEL
6: JOYAS
7: MONEDAS
```

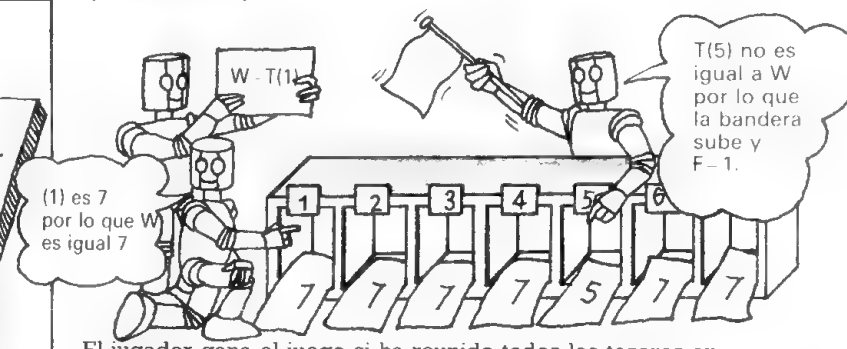
Primero comprueba el valor de C para saber si el jugador lleva algo. Si lo lleva manda a la computadora a un bucle que busque qué elemento de la matriz T vale 999, para a continuación escribir el nombre de la caja del tesoro que lleva buscando el elemento de la matriz T$ que tiene el mismo «número de compartimiento». Para escribir los contenidos de las habitaciones necesitarás otro bucle que escriba la posición de los tesoros en ese momento exceptuando el que lleva el jugador (si lo lleva).

10. Cuenta las jugadas (paso 600)



En el paso 600, suma 1 a la variable M para llevar la cuenta de las jugadas que se han realizado.

11. ¿Están todos los tesoros en la misma habitación? (Pasos 610-690)



El jugador gana el juego si ha reunido todos los tesoros en una misma habitación. La manera de saberlo es comprobar los números almacenados en la matriz T, ya que serán todos iguales cuando todos los tesoros están en una misma habitación. Para hacerlo toma un elemento de T, T(1) por ejemplo, y almacénalo en una variable temporal que llamaremos W. A continuación compárala mediante un bucle con todos los demás números almacenados en T. Utiliza con todos los demás números almacenados en T. Utiliza una variable bandera para indicar si son o no todos los números iguales.

12. ¿Ha gastado todas sus jugadas? (Pasos 700-750)

Decide el límite de jugadas de que puede disponer el jugador.Si M es mayor que este número, entonces el jugador pierde. Si es éste el caso escribe un mensaje y para el programa.

13. ¿Ha cambiado el jugador de posición? (pasos 800-85-)

```
¿QUE QUIERES HACER?

CONTINUAS EN LA HABITACION 7
```

Si el jugador introdujo N, E, S u O en A$ en el paso INPUT, mándale de vuelta al paso 400 para identificar la nueva habitación y sus contenidos. Si no lo ha hecho, recuérdale su posición y haz que la computadora vuelva al paso INPUT (500).

Solución a los problemas

En las páginas siguientes encontrarás todas las respuestas de los problemas y proyectos de este libro. Los programas están escritos en BASIC standard, por lo que quizás tengas que convertir algunas instrucciones no estandarizadas como RND o CLS para que funcionen en tu computadora. Si alguno de los programas no te funciona, comprueba que tu computadora admite todas las instrucciones BASIC del programa. En la página 47 encontrarás una tabla de conversión que te puede ser útil. Los pasos que necesitan ser alterados completamente para que valgan para las computadoras Sinclair (Timex) están señalados con △ dándo de los pasos válidos a continuación. Los pasos que sólo han de ser cambiados para el ZX81 (Timex 1000) están señalados con ▲.

En ocasiones verás que los programas que tú has escrito son diferentes a los que se ofrecen en las soluciones. Esto no importa siempre que tu programa funcione bien. Sin embargo, estudia la solución y procura que el programa sea lo más corto y claro posible.

Conociendo el BASIC (páginas 4-5)

Programas simples

1
```
40 PRINT «HOLA»;
50 PRINT A$
```

Este punto y coma hace que la computadora continúe en la misma línea para escribir **A$**.

2
```
40 PRINT «HOLA», A$
50 BORRA ESTE PASO
```

La coma hace que la computadora deje algunos espacios antes de escribir **A$**.

3
```
40 PRINT TAB(6); «HOLA»; A$
50 BORRA ESTE PASO
```

Algunas computadoras no necesitan punto y coma detrás de TAB.

4
```
40 PRINT «          ADIOS»
50 PRINT «          »; A$
```

Problemas de variables (páginas 6-7)

Elección de nombres de variables

Estos nombres contienen palabras en BASIC y no pueden usarse como nombres de variables: **LETRA$** (LET); **RUN$** (RUN). El ZX81 (Timex 1000) sólo admitirá nombres de variables de cadena de una sola letra, por lo que tampoco puedes usar **MOSCA$**.

Problema con PRINT

```
10 LET A=66
20 LET B=77
▲30 LET RR$=«ROBOTS OXIDADOS»
40 PRINT A; «  »; RR$; «COMIERON»
50 PRINT B; «SALCHICHAS CHAMUSCADAS»
```

Un espacio vacío entre comillas hace que la computadora deje un espacio en blanco.

Fíjate en un programa

```
45 PRINT «C IS»; C
55 PRINT «D IS»; D
```

Grados centígrados a Farenheit

```
40 LET C=9/B
50 LET D=C*A
60 LET R=D+F
```

Cambia los pasos 40-60 como se indica aquí.

Problema: escribir un programa

1
```
10 LET U$=«UG»
20 PRINT «QUE PALABRA»
30 INPUT W$
40 PRINT «EN UGIANO ES»;
50 PRINT U$; W$
```

El paso 50 hace que la computadora escriba UG seguido de la palabra almacenada en **W$**.

2
```
10 PRINT «CUAL ES LA DISTANCIA»;
20 INPUT D
30 PRINT «CUANTO TARDASTE»;
40 INPUT T
50 LET S=D/T
60 PRINT «TU VELOCIDAD ERA DE»; S;
70 PRINT «MILLAS POR HORA»
```

En el paso 50, la computadora calcula la velocidad y la almacena en **S***.

3
```
10 LET A=30
20 LET B=20
30 LET S=35
40 LET T=S/B
50 LET C=A*T
60 LET D=C+S
70 PRINT «DEBEN COMPRAR»;
80 PRINT D; «SALCHICHAS.»;
90 PRINT «TARDARAN»;
100 PRINT T; «HORAS»
```

A y B son las salchichas que pueden comer los robots en una hora. S son las salchichas que quiere comer el robot 2 y T el tiempo que tarda el robot 2 en comer sus salchichas (S). El paso 50 calcula el número de salchichas que el robot 1 come en ese tiempo (T) y D es el número total de salchichas.

Repetición de cosas (páginas 8-9)

Problema: Bucles

1
```
10 FOR J=1 TO 100
20 PRINT «HOLA»;
30 NEXT J
```

El punto y coma del paso 30 hace que la computadora permanezca en el mismo paso para escribir la siguiente palabra.

2
```
10 FOR J=1 TO 25
20 PRINT TAB(15); «HOLA»
30 NEXT J
```

Así la computadora escribe una columna de 25 HOLAs a 15 espacios de la parte izquierda de la pantalla.

*En algunas computadoras (ej.: Vic 20 y Pet) no hace falta el punto y coma al final de los pasos PRINT antes de INPUT.

3 El paso 20 está mal, ya que afecta al contador de bucle, I. Cada vez que se ejecuta el bucle, el paso 20 cambia el valor de I, de 1 a 0.

Problemas con step

1
```
10 FOR I=25 TO 1 STEP -1
20 PRINT TAB(I); «HOLA»
30 NEXT I
```

Utiliza la variable del bucle (I) como tabulador. Asegúrate de que I no sea mayor que la anchura de tu pantalla. Cada vez que se ejecute el bucle a I se le resta 1.

2
```
10 FOR L=5 TO 0 STEP-1
20 PRINT TAB(5); L; TAB(10); L*L
30 NEXT L
```

Step-1 hace que la variable del bucle cuente al revés de 5 a 0. Cada vez que se ejecuta el paso 20 escribe el valor de L y luego lo eleva al cuadrado.

3
```
10 PRINT «INICIO»;
20 INPUT A
30 PRINT «FINAL»;
40 INPUT B
50 PRINT «SALTAR»;
60 INPUT C
70 FOR J=A TO B STEP C
80 PRINT J; " ";
90 NEXT J
```

El bucle de los pasos 70-90 utiliza A para asignar el valor inicial de la variable del bucle, B para asignar el valor final y C para asignar los saltos (step).

Mensaje secreto

```
10 PRINT «MENSAJE SECRETO»
20 PRINT «MEMORIA EN 5 SEGUNDOS»
30 PRINT «LUEGO DESAPARECERA»
40 PRINT
50 PRINT «REUNIRSE CON AGENTE X. 2.00 AEROPUERTO»
60 FOR I=1 TO 1000          Bucle de retardamiento
70 NEXT I
80 CLS                      Borra la pantalla
```

Programa para dibujos

2
```
50 PRINT TAB(A-K); «**»;
60 PRINT TAB(A+K); «*»
70 NEXT K
```

A es el vértice superior del programa. Para dibujar la parte izquierda, restas K de A y para la derecha sumas K a A.

3
```
30 BORRA ESTE PASO
40 FOR K=9 TO 1 STEP -1
80 PRINT TAB(A); «*»
```

Cambia estos pasos para que aparezca la figura boca abajo. Para realizar los demás dibujos de la página 9 usa el mismo programa con diferentes posiciones TAB.

Problemas de bucles (páginas 10-11)

Bucle para hacer que el programa vaya más lento

Podrías añadir este bucle anidado en el paso 45 o en el 55.

```
45 FOR K=1 TO 1000
48 NEXT K
```

Chinches en los bucles

Los pasos 50 y 70 están mal; deberían ser así:

```
50 NEXT K
70 NEXT I
```

Contador binario

Para hacer un programa que cuente números de ocho cifras necesitarás cuatro bucles más (E, F, G, H). Tendrás que volver a numerar el programa y alterar los pasos PRINT de esta manera:

```
PRINT H-G*2-F*4-E*8-D*16-C*32-B*64-
A*128; «=»;
PRINT A;B;C;D;E;F;G;H
```

Mensaje intermitente

```
10 FOR J=1 TO 10
20 CLS
30 FOR K=1 TO 1000: NEXT K
40 PRINT TAB(10); «PELIGRO»
50 PRINT TAB(7); «ATAQUE ESPACIAL»
60 FOR K=1 TO 1000: NEXT K
70 NEXT J
```

Este es un programa para que aparezca un mensaje intermitente. Tendrás que cambiar el tamaño del bucle de retardamiento, así como las posiciones TAB para que se adapte a tu pantalla. Si tu computadora no admite pasos con varias instrucciones, pon la instrucción NEXT K en el paso siguiente.

Reloj de la computadora

```
10 FOR J=0 TO 59                    Bucle del minutero
20 FOR K=0 TO 59
30 PRINT J; «:»; K
40 FOR L=1 TO 500: NEXT L           Bucle del segundero
50 CLS
60 NEXT K
70 NEXT J
```

Haz que el bucle K se ejecute en 1 segundo.

Despegue del cohete

```
 10 CLS
▲20 FOR I=1 TO 20
 30 PRINT
 40 NEXT I
 50 PRINT "  *  "
 60 PRINT " *** "
 70 PRINT " *** "
 80 PRINT " *** "
 90 PRINT "*** ***"
100 FOR J=1 TO 25
▲110 PRINT
120 FOR K=1 TO 1000                 Bucle de retardamiento
130 NEXT K
140 NEXT J
```

El bucle de los pasos 20-40 hace que la computadora deje líneas en blanco para que el cohete aparezca en la parte inferior de la pantalla. Los pasos 50-90 dibujan el cohete. Debes hacer que el bucle de los pasos 100-140 se repita tantas veces como el número de líneas de tu pantalla. Cada vez que se ejecute el bucle la computadora deja una línea en blanco y el cohete sube una línea hacia arriba.

Cambios para Sinclair (Timex)

Para ejecutar el programa en el Spectrum (Timex 2000) aprieta ENTER cuando aparezca el mensaje scroll en la pantalla.

▲Para el ZX81 (Timex 1000) cambia estos pasos.

```
20 FOR I=1 TO 17
110 SCROLL
```

Programa del saltador

```
10 FOR J=1 TO 25 STEP 2
20 CLS
30 PRINT
40 PRINT TAB(J); «o»
50 PRINT TAB(J); «<O>»
60 PRINT TAB(J); «/ \»
70 FOR K=1 TO 1000
80 NEXT K
90 CLS
100 PRINT TAB(J+1); «<o>»
110 PRINT TAB(J+1); «O»
120 PRINT TAB(J+1); «/ \»
130 FOR K=1 TO 1000
140 NEXT K
150 NEXT J
```

Haz que el bucle de los pasos 10-150 se ejecute tantas veces como el número de caracteres que caben a lo ancho de tu pantalla. Los pasos 40-60 dibujan al hombre en la posición inicial. Los pasos 70-80 y 130-140 son bucles de retardamiento inicial. Los pasos 70-80 y 130-140 son bucles de retardamiento que hacen posible que el hombre permanezca en la pantalla un momento. Los pasos 100-120 dibujan al hombre en la segunda posición.

Ejercicios con IF/THEN (páginas 12-13)

Tablas de multiplicar

```
55 IF B<>J*A THEN PRINT «MAL»;
J; «X»; A;;«=»; J*A
```

Añade este paso para que la computadora te diga si tu respuesta está mal.

Contraseña

```
50 IF P$<>S$ THEN PRINT «MAL.
CABEZOTA»
60 IF P$=SS THEN PRINT «OK CONTINUA»
```

Añade estos pasos (con tus propios mensajes) para completar el programa de la contraseña.

```
10 LET S$=«SALCHICHAS»
20 LET N=007
30 PRINT «CONTRASEÑA, POR FAVOR»;
40 INPUT P$
50 PRINT «NUMERO SECRETO»;
60 INPUT SN
70 IF P$<>S$ OR SN<>N THEN PRINT
«MAL. CABEZOTA»
80 IF P$=S$ AND SN=N THEN PRINT
«OK. CONTINUA»
```

Cambia el programa de la computadora de esta manera para que la computadora te pida también un número secreto.

Calculadora en la computadora

```
10 PRINT «PIENSA EN UN NUMERO»;
20 INPUT X
30 PRINT «Y OTRO»;
40 INPUT Y
50 PRINT «QUIERES:»
60 PRINT «SUMAR, RESTAR, DIVIDIR»
70 PRINT «O MULTIPLICAR?»
80 INPUT A$
90 IF A$=«SUMAR» THEN LET A=X+Y
100 IF A$=«RESTAR» THEN LET A=X-Y
110 IF A$=«DIVIDIR» THEN LET A=X/Y
120 IF A$=«MULTIPLICAR» THEN LET A=X*Y
130 PRINT «LA RESPUESTA ES»; A
```

La computadora realiza una operación diferente según la respuesta que introduces en **A$**.

Adivinanzas

```
45 IF Y<X THEN PRINT «DEMASIADO PEQUEÑO»
46 IF Y>X THEN PRINT «DEMASIADO GRANDE»
```

Añade estos pasos para que la computadora te diga si tu número es demasiado grande o demasiado pequeño.

```
5 LET N=0
55 LET N=N+1
56 IF N>5 THEN STOP
```

Añade estos pasos para contar el número de intentos y para el programa después de cinco.

Juego con palabras

```
10 PRINT «PALABRA, POR FAVOR»
20 INPUT W$
30 PRINT «PISTA, POR FAVOR»
40 INPUT C$
50 CLS
60 PRINT «PISTA:»
70 PRINT C$
80 PRINT «ADIVINA LA PALABRA»
90 INPUT G$
100 IF (G1=H1 AND G2<>H2) OR (G2=H2
110 PRINT «NO»
120 GOTO 90
130 PRINT «SI»
```

Los pasos 10-40 hacen que la computadora te pida una palabra y una pista.
Los pasos 50-60 borran la pantalla y escriben la pista.
El paso 120 manda la computadora de vuelta al paso 90 para otro intento.

Carrera de caballos

Estos son los pasos IF/THEN completados para la Carrera de Caballos.

```
80 IF G1=H1 AND G2=H2 THEN GOTO 160
90 IF G1=H2 OR G2=H1 THEN GOTO 150
100 IF (G1=H1 AND G2<>H2) OR (G2=H2
AND G1<>H1) THEN GOTO 140
120 IF N=4 THEN GOTO 170
```

Paso 80: Debes mandar a la computadora al paso 160, si los dos pronósticos son ciertos.
Paso 90: Si el caballo es correcto pero la posición está mal, manda a la computadora al paso 150.
Paso 100: Si uno de los dos pronósticos es correcto, ve al paso 140.
Paso 120: Tras cuatro intentos, manda a la computadora al paso 170.

Ideas para mejorar el juego

```
15 LET S=0
140 PRINT «UN PRONOSTICO CORRECTO»: LET
S=2: GOTO 120
160 PRINT «CORRECTO:» LET S=4
180 PRINT «TU PUNTUACION ES»; S
```

Esta es una idea para un sistema de puntuación que da cuatro puntos por adivinar los dos caballos y dos puntos por tener uno bien. La variable S contiene la puntuación.

```
190 PRINT «¿QUIERES VOLVER A JUGAR? (Y/N)»
200 INPUT Z$
210 IF Z$=«Y» THEN GOTO 10
220 IF Z$=«N» THEN STOP
```

Para dar la ocasión al jugador de volver a jugar, introduces estos pasos.

Números aleatorios (páginas 14-15)

Números aleatorios entre el 10 y el 20

```
INT(RND(1)*11+10)
```

Para hacer esto, multiplica por la cantidad de números posibles (11) y a continuación suma el primer número posible (10). Utiliza la instrucción RND de tu computadora.

Problema: alterar un programa

```
20 LET X=INT (RND(1)*20+1)
```

Cambia el paso 20 del juego de acertar números para hacer que la computadora escoja un número aleatorio entre 1 y 20.

Secuencia de números

Añade estos pasos al programa para tener un juego de secuencia de números.

```
  5 PRINT «ACIERTA EL NUMERO SIGUIENTE»
  7 PRINT «DE ESTA SECUENCIA»
 10 LET X = INT(RND(1)* 10+)
 20 LET Y=INT(RND(1)* 10+1)
 60 LET A  X+4*Y
 70 INPUT N
 80 IF N = A THEN PRINT «CORRECTO»
 90 IF N<>A THEN PRINT «MAL. ES EL»; A
100 GOTO 5
```

El paso 60 calcula el siguiente número de la secuencia.

Secuencia aleatoria

```
24 LET R=INT(RND(1)*3 - 1)
25 IF R=1 THEN GOTO 30
26 IF R=2 THEN GOTO 40
27 IF R=3 THEN GOTO 50
```

Para seleccionar una secuencia aleatoriamente haz que la computadora escoja un número entre el 1 y 3 aleatoriamente y lo almacene en R. Manda a la computadora a una secuencia diferente según sea el número almacenado en R.

```
30 FOR I=1 TO 3
33 PRINT X + I*I
35 NEXT I
37 LET A=X+4*4
39 GOTO 70

40 FOR I=1 TO 3
43 PRINT I*I - Y
45 NEXT I
47 LET A=4*4  Y
49 GOTO 70

50 FOR I=1 TO 3
53 PRINT X+Y  I*I
55 NEXT I
57 LET A=X+Y-4*4
```

Para añadir tres secuencias diferentes de números al programa, borra los pasos 30-60 y añade un bucle para cada secuencia. Al final de las dos primeras secuencias manda a la computadora al paso 70.

Fuga de Zorgos

```
 10 LET C=10                                  C Lleva la cuenta del número de chips.
 20 PRINT «TIENES»; C; «CHIPS»
 30 PRINT «COLOCA TU APUESTA:»;
 40 INPUT B
 50 IF B>C THEN PRINT «NO TIENES              Ten presente que la apuesta no puede ser mayor
 TANTOS CHIPS»: GOTO 20                       que el número de chips.
 60 LET C=C-B                                 Calcula cuántos chips quedan.
 70 LET X=INT(RND(1)*6+1)                     Escoge dos números aleatorios entre el 1 y el 6.
 80 LET Y=INT(RND(1)*6+1)
 90 PRINT «APRIETA P PARA TIRAR:»;
100 INPUT P$
110 IF P$<>«P» THEN GOTO 90                   Comprueba lo que introduce el jugador.
120 PRINT TAB(5); X; TAB(10); Y               Escribe los números elegidos en la pantalla.
130 IF X=Y THEN GOTO 250
140 IF X+Y=10 OR X+Y=11 THEN GOTO 210         Comprueba los números y manda a la
150 IF X+Y=6 OR X+Y=7 THEN GOTO 190           computadora a diferentes pasos para que calcule
160 PRINT «LO SIENTO. PIERDES LO APOSTADO»    la puntuación.
170 IF C=0 THEN GOTO 290                      Comprueba el número de chips que quedan. Si
                                              C = 0, manda a la computadora al final del
                                              programa.
180 GOTO 20                                   Hace que la computadora vuelva a por otra
190 PRINT «NO PASA NADA. CONSERVAS LA APUESTA» apuesta.
200 LET C=C+B: GOTO 20                        Ajusta el valor de C.
210 PRINT «BIEN HECHO. TRIPLICAS LA APUESTA»
220 LET C=C+(B*3)
230 IF C>=50 THEN GOTO 320                    Si C es 50 o más, manda a la computadora al
240 GOTO 20                                   final del programa.
250 PRINT «DUPLICAR LA APUESTA»
260 LET C=C+(B*2)
270 IF C>=50 THEN GOTO 320
280 GOTO 20
290 PRINT «LO SIENTO. TE HAS QUEDADO SIN CHIPS»
300 PRINT «NO HAY ESCAPE DE ZORGOS»
310 STOP
320 PRINT «BIEN HECHO. AHORA PUEDES ESCAPAR»
330 PRINT «DE ZORGOS CON»; C; «CHIPS»
```

Papel, piedra o tijeras: localiza los errores

Corrige estos pasos como se indica a continuación para hacer que el programa funcione bien.

```
 50 LET R=INT(RND(1)*3+1)
 80 IF R=3 THEN LET C$="TIJERAS"
150 IF C$="TIJERAS" AND A$="PIEDRA"
THEN LET F=1
190 PRINT "POR TANTO ";
210 IF F=1 THEN PRINT "TU GANAS"
230 IF F=0 THEN LET C=C+1
240 IF F=1 THEN LET A=A+1
280 IF C<10 AND A<10 THEN GOTO 40
```

Jugar con caracteres (páginas 16-17)

Problemas con cadenas

1 △ `40 PRINT TAB(K);MID$(N$,K,1)`

Esto hace que la computadora escriba de una en una las letras de la palabra en la posición de TAB K, comenzando con la letra número K.

2
```
 10 LET K$="CANGURO"
 20 LET L=LEN(K$)
 30 FOR J=L TO 1 STEP -1
△40 PRINT MID$(K$,J,1);
 50 NEXT J
```

Utiliza un bucle con step-1 para hacer que la computadora escriba una palabra comenzando por detrás.

3 △ `50 PRINT RIGHT$(S$,L-J)+LEFT$(S$,J)`

Cada vez que se ejecuta el bucle, RIGHT$(S$,L-J) hace que la computadora escriba L (la longitud de la palabra) menos J letras desde la derecha del círculo, y luego añade J letras desde la izquierda. Intenta ejecutar este programa con distintas palabras.

La palabra más larga

```
△ 50 IF LEN(W$)<LEN(A$)
  THEN LET A$=W$
```

Cada vez que se ejecuta el bucle, la computadora compara la longitud de W$ con A$. Si W$ es más largo que A$, sustituye A$ por W$ y pasa a comparar la siguiente palabra. Al final, A$ tendrá la palabra más larga de las introducidas.

Palabra más corta

```
10 LET A$="XXX!!!&&ABC***123!!!!
XXXXXXXXXX"
50 IF LEN(W$)<LEN(A$)
THEN LET A$=W$
70 PRINT "PALABRA MAS CORTA"
```

Este programa es igual al de la palabra más larga, excepto en los pasos 10 y 50. El paso 10 contiene una cadena de caracteres para que la computadora compare la longitud de cada palabra con ella. No importa qué caracteres introduzcas en A$.

Editor de palabras

Estos son los pasos completados del editor de palabras.

```
40 LET S$=" "+S$+" "
70 LET W$=" "+W$+" "
```

Tienes que añadir espacios a ambos lados de S$ y W$ para estar seguro en el paso 140 de que la computadora sólo busca palabras. Sin los espacios, la computadora tomaría cualquier caracter de la frase que coincidiese con S$ o W$. Intenta ejecutar el programa sin dejar espacios para ver lo que sucede.

```
100 LET LS=LEN(S$)
110 LET LW=LEN(W$)
```

Estos pasos hacen que LS valga lo mismo que la longitud de S$ y LW lo mismo que la longitud de W$. El paso 100 se vuelve a repetir en el 160 para hacer que LS valga la longitud de la nueva frase.

```
△ 140 IF MID$(S$,K,LW)=W$
  THEN LET A$=S$
```

Cada vez que se repite el bucle MID$ (S$, K, LW) hace que la computadora revise LW (la longitud de W$) caracteres de S$, comenzando en el caracter K. Cuando encuentra una secuencia de caracteres que coinciden con W$ almacena la frase completa en una nueva variable A$.

```
△ 150 IF A$=S$ THEN LET S$=LEFT$(A$,K)
  +N$+RIGHT$(A$,LS  (K+LW-2))
```

LEFT$(A$,K) saca el número de caracteres a la izquierda de la palabra que piensas reemplazar. La mejor forma de entender cómo funciona esta línea es sacándola con una frase en un trozo de papel.

```
180 IF K<=LS-LW+1 THEN GOTO 140
```

Manda a la computadora de vuelta al paso 140 para revisar el resto de la frase si la palabra que quieres sustituir se halla más de una vez.

Cambiar pasos a las computadoras Sinclair (Timex)

Problemas con cadenas

Incluye estos pasos en los Problemas con Cadenas: soluciones 1, 2 y 3.

1 `40 PRINT TAB(K);N$(K TO K)`

2 `40 PRINT K$(J TO J);`

3 `50 PRINT S$(J+1 TO )+S$( TO J)`

Editor de palabras

```
140 IF S$(K TO K+LW  1)=W$ THEN
LET A$=S$
```

```
150 IF A$=S$ THEN LET S$=A$( TO K)
+N$+A$(K+LW  1 TO )
```

Sustituye estos pasos en el programa Editor de palabras.

Más operaciones con caracteres (página 18)

Problemas con letras

1 ▲10 FOR K=65 TO 90

Estos son los números ASCII para escribir el alfabeto. En el ZX81 (Timex 1000) los números son 38 a 63.

2
```
10 FOR K=97 TO 122
20 PRINT CHR$(K);
30 NEXT K
```

Estos son los números para escribir el alfabeto en minúsculas (En algunas computadoras como el Dragón que no utiliza minúsculas, estos números se utilizan para un grupo de mayúsculas diferentes).

3 ▲
```
10 LET R=INT(RND(1)*26+65)
20 PRINT CHR$(R);
30 GOTO 10
```

Esta es la forma más sencilla de escribir una serie de letras aleatorias. Para el ZX81 (Timex 1000) sustituye 65, el número de código de la primera letra del alfabeto, por 38.

Compara letras

```
10 INPUT X$,Y$
20 IF X$<Y$ THEN PRINT X$; «VIENE ANTES
QUE; Y$
30 IF Y$<X$ THEN PRINT Y$; «VIENE ANTES
QUE »;X$
40 GOTO 10
```

Si tu computadora sólo admite una variable detrás de INPUT, utiliza dos pasos INPUT.

Transformar en mayúsculas

```
50 IF X$>=«a» AND X$<=«z» THEN PRINT
CHR$(ASC(X$)-32);
60 IF X$>=«A» AND X$<=«Z» THEN PRINT
CHR$(ASC(X$)-32);
```

El número que falta en los pasos 50 y 60 es el 32. Esta es la diferencia en el número de código de las mayúsculas y el de las minúsculas. Hecha un vistazo a la tabla de números de código de tu manual.

Programas para escribir en clave (página 19)

Programa para escribir en clave

Este es el programa completo con los números y símbolos que faltaban. Utiliza este programa para el Spectrum (Timex 2000) pero cambia la instrucción ASC por CODE. Para el ZX81 (Timex 1000) utiliza el programa de la derecha.

```
10 PRINT «CUAL ES TU MENSAJE»
20 INPUT M$
30 FOR J=1 TO LEN(M$)
△ 40 LET X=ASC(M!D$(M$,J,1))
50 IF X<65 OR X>90
THEN LET N=X: GOTO 100
60 IF INT(J/2)=J/2 THEN LET N=X+1
70 IF INT(J/2)<>J/2 THEN LET N=X-1
80 IF N<65 THEN LET N=N+26
90 IF N>90 THEN LET N=N-26
100 PRINT CHR$(N);
110 NEXT J
```

▲Programa para escribir en clave en el ZX81

```
10 PRINT «CUAL ES TU MENSAJE»
20 INPUT M$
30 FOR J=1 TO LEN(M$)
40 LET X=CODE(M$(J TO J))
50 IF X<38 OR X>63 THEN LET N=X
55 IF X<38 OR X>63 THEN GOTO 100
60 IF INT(J/2)=J/2 THEN LET N=X+1
70 IF INT(J/2)<>J/2 THEN LET N=X-1
80 IF N<38 THEN LET N=N+26
90 IF N>63 THEN LET N=N-26
100 PRINT CHR$(N);
110 NEXT J
```

Los pasos 60-70 comprueban si la variable del bucle J es par o impar dividiendo entre 2 y utilizando INT para convertir el resultado en un número entero. A continuación la computadora comprueba si la respuesta es igual a J/2 y si no lo es J es un número par.

Número en clave

Para este programa utiliza el anterior añadiéndole unos pasos que hagan posible el introducir un número de clave. También tendrás que cambiar el paso 60 para sumar el número clave (K) al número de código ASCII (X) de cada caracter, así como borrar el paso 70.

```
25 PRINT «¿CUAL ES EL NUMERO DE CLAVE?»
27 INPUT K
60 LET N=X+K
70 BORRA ESTE PASO
```

Clave de bucle

Este es también el Programa para Escribir en Clave pero haciendo que en el paso 60 se le sume la variable del bucle (J) al número de código ASCII de cada carácter.

```
60 LET N=X+J
70 BORRA ESTE PASO
```

Descodificador de clave de bucle

Para escribir un descodificador cambia el paso 60 a:

```
60 LET N=X-J
```

Clave inversa

```
10 INPUT «MENSAJE »;M$
20 FOR J=1 TO LEN(M$) STEP 2
△ 30 PRINT MID$(M$,J+1,1);
△ 40 PRINT MID$(M$,J,1);
50 NEXT J
```

Step 2 hace que la computadora cuente de dos en dos. Cada vez que se repite el bucle escribe el segundo caracter (J + 1) de cada pareja en la pantalla, seguido del primer caracter (J).

Cambios para hacer la clave inversa en Sinclair (Timex)

```
30 PRINT M$(J+1 TO J+1);
40 PRINT M$(J TO J);
```

Ejercicios con INKEY$ (páginas 20-21)

Escribe HOLA

```
10 LET A$=INKEY$
20 IF A$="" THEN PRINT " ";
30 IF A$<>" " THEN PRINT "HOLA";
40 GOTO 10
```

Si no se aprieta ninguna tecla, A$ está vacía y la computadora deja un espacio en blanco (paso 20). Si se aprieta una tecla, se almacena en A$ y la computadora escribe HOLA (paso 30).

Haz que la computadora espere

```
10 LET A$=INKEY$
20 IF A$="" THEN GOTO 10
30 RESTO DEL PROGRAMA...
```

El paso 20 hace que la computadora repita el paso 10 siempre que no se aprieta ninguna tecla.

Matemáticas rápidas con chinches

Corrige los pasos como se indica a continuación para hacer que el programa funcione bien.

```
20 PRINT "APRIETA CUALQUIER TECLA CUANDO"
50 LET X=INT(RND(1)*25+1)
80 PRINT X;" + ";Y;" = "
110 LET A$=INKEY$
140 IF N<50 THEN GOTO 90
160 FOR K=1 TO 1000:NEXT K
170 GOTO 40
190 PRINT "SI, LA RESPUESTA ES"; X+Y
```

- Pon las comillas que faltan.
- Multiplica por 25 y suma 1 para obtener un número aleatorio entre el 1 y el 15.
- Quita las comillas que hay delante de la X.
- El nombre de la variable debe ir antes que INKEY$.
- N debe ir hasta 50 para dar cabida a todas las respuestas posibles de X + Y.
- Variable incorrecta detrás de NEXT.
- Cambia el número de paso para que N vuelva a valer 0.
- Coloca los nombres de las variables fuera de las comillas.

Juego de coches de choque (página 21)

Este es el programa para los coches de choque.

```
10 CLS
20 LET C=5
30 LET I=1
40 LET A=10
50 LET D=I+A
60 IF I=1 THEN LET N=1
70 IF I=25 THEN LET N=0
80 IF N=1 THEN LET I=I+1
90 IF N=0 THEN LET I=I-1
100 LET A$=INKEY$
110 IF A$="" THEN LET C=C-1
120 IF A$="" THEN LET C=C+1
130 PRINT TAB(L);"!";TAB(C);"**";TAB(D);"!"
140 IF C<=L OR C>=R THEN PRINT "***CRASH8 ***"
150 GOTO 50
```

- Cuando I vale lo que los límites de la pantalla, N cambia de valor.
- Cuando N es 1, el valor de I aumenta en uno. Cuando N es cero, el valor de I disminuye uno.
- Estos pasos te permiten conducir el coche.
- Dibuja la carretera y el coche.
- Calcula si el coche se ha estrellado.
- Manda de vuelta a la computadora al paso 50 para borrar el valor de D cada vez que I cambia.

Añade este bucle de retardamiento si el programa va demasiado aprisa.

```
142 FOR K=1 TO 400:NEXT K
```

Sistema de puntuación

```
140 IF C<=L OR C>=R THEN GOTO 160
145 IF A$<>" " THEN LET P=P+1
160 PRINT "***CRASH***"
170 LET CR=CR+1
180 IF CR<=5 THEN GOTO 20
190 PRINT "FIN DEL JUEGO"
200 PRINT "TU PUNTUACION ES ";S
210 STOP
```

El paso 145 te da un punto cada vez que mueves el coche sin chocarte. El paso 180 para el programa después de cinco choques. Debes dar a CR (número de choques) y a P (la puntuación) un valor inicial de 0 al comenzar el programa.

Hacer que la carretera tenga curvas

```
75 IF I=5 OR I=10 THEN
LET N=INT(RND(1)*2)
76 IF I=15 OR I=20 THEN
LET N=INT(RND(1)*2)
```

Una manera de hacer esto es dejar que la computadora elija aleatoriamente en qué dirección ir cuando I tome determinados valores. Para hacer esto tienes que cambiar el valor de N aleatoriamente cuando I valga por ejemplo 5, 10. 15 y 20.

Problemas con DATA (páginas 22-23)

Comprobación de nombres

```
50 IF X$«JIM» THEN PRINT «TU NOMBRE NO
ESTA EN LA LISTA»:STOP
80 DATA CHARLIE, JEMIMA, DICK EL TUERTO
90 DATA SAMPSON,DELILAH,JIM
```

Pon tu propia lista de nombres en pasos DATA como éstos teniendo cuidado de poner comas entre los nombres. Pon el último nombre de la lista en el paso 50. Si tu computadora no admite pasos con varias instrucciones, repite la instrucción IF/THEN con STOP en un nuevo paso.

Volver DATA al principio

```
10 FOR J=65 TO 90                    Códigos ASCII
20 FOR I= 1 TO 19                    Número de datos
△40 IF LEFT$(N$,1) =CHR$(J) THEN PRINT N$
```

Cambios para Sinclair (Timex)

```
40 IF N$(1 TO 1)=CHR$(J) THEN PRINT N$
```

Localiza el chinche

1. Los datos contienen símbolos como – y /, por lo que en el paso 20 debes cambiar la variable por una variable de cadena, como N$.

2. La instrucción GOTO hace que la computadora continúe leyendo los datos después de haber llegado al final de la lista de datos, por lo que obtendrás un error del tipo «OUT OF DATA AT LINE 50» (sin suficientes datos en el paso 50). Para resolver esto puedes usar un bucle que se repita tantas veces como datos haya o poner el último dato en un paso IF/THEN como en el comprobador de nombres.

El café de Joe

```
10 PRINT «BIENVENIDOS AL CAFE DE JOE»
20 PRINT «¿CUANTO QUIERE GASTAR?»
30 INPUT X
40 PRINT «ESTO ES LO QUE PUEDES TOMAR»
50 PRINT
60 READ Y,Y$
70 IF Y<=X THEN PRINT Y$
80 IF Y$=«BATIDO DE MENTA» THEN STOP
90 GOTO 60
100 DATA 1.99,GUISO DE CARACOLES A LA FRANCESA
```

Lista todos los otros precios y datos del menú en pasos DATA terminando con BATIDO DE MENTA.

Agenda de teléfono

```
10 PRINT «QUE TELEFONO»
20 PRINT «QUIERES»
30 PRINT
40 INPUT N$
50 READ X$,Y$ }
60 IF N$=X$ THEN GOTO 90 }
70 IF X$=« » THEN PRINT «NOMBRE NO LOCALIZADO» }
80 GOTO 50
90 PRINT X$;«:»;Y$
100 PRINT «¿QUIERES ALGUN OTRO NUMERO?»
110 INPUT A$
120 IF A$=«SI» THEN GOTO 150
130 IF A$=«NO» THEN STOP
140 PRINT «NO ENTIENDO»:GOTO 110 }
150 RESTORE:GOTO 10 }
160 DATA ARDILLA,670-5054
170 DATA ROBOT OXIDADO,60-14-444 }
```

Lee nombres almacenándolos en X$ y números en Y$.

Va al paso 90 cuando encuentra el nombre.

Pon el último nombre de tu lista entre las comillas de este paso.

Necesitas este paso por si en el paso 110 escribes algo que no sea sí o no.

Manda a la computadora al principio de la lista de datos.

Lista nombres y números en pasos DATA como éstos.

Uso de matrices (páginas 24-25)

Matrices numéricas

```
10 DIM N(6)
50 DATA 1066,1216,1485,1603,1665,1959
```

Escribir los datos en la pantalla

Necesitarás un bucle como éste para escribir los datos en la pantalla (acuérdate de renumerar el paso DATA).

```
50 FOR K=1 TO 6
60 PRINT «N(«;K;») IS »;N(K)
70 NEXT K
```

Este bucle escribe los elementos de la matriz aleatoriamente.

```
50 FOR K=1 TO 10
60 LET R=INT (RND(1)*6+1)
70 PRINT N(R),
80 NEXT K
```

Matrices de caracteres

```
△10 DIM N$(5)
  20 FOR I=1 TO 5
▲30 READ N$(I)
  40 NEXT I
  50 FOR I=1 TO 5
  60 PRINT «N$(«;I;») IS »;N$(I)
  70 NEXT I
  80 DATA TONY,CORINNE,JULIA,CHRIS,GABY
```

Cambios para Sinclair (Timex)

```
  10 DIM N$(5,7)
▲ 30 INPUT N$(1)
```

Calculador de meses

Completa los pasos que faltan y las variables de la siguiente manera:

```
△10 DIM M$(12),D(12)
▲30 READ M$(K),D(K)
  70 PRINT M$(N);« TIENE »;
  80 PRINT D(N);« DIAS »
```

Añade el resto de los pasos data de esta forma:

```
▲90 DATA ENERO,31
```

Incluye estos pasos para alterar el programa.

```
  50 PRINT «QUE MES»
  60 INPUT A$
  62 FOR I= 1 TO 12
△63 IF M$(I)=A$ THEN LET F=I
  64 NEXT I
  70 PRINT A$;« TIENE »;
  80 PRINT D(F);» DIAS »
```

Cambios para Sinclair (Timex)

```
  10 DIM M$(12,9)
  15 DIM D(12)
▲ 30 INPUT M$(K)
▲ 35 INPUT D(K)

  63 IF M$(I) ( TO LEN(A$))= A$ THEN
  LET F= 1
```

Programa para decidir qué hacer

```
△ 10 DIM I$(10)
  20 FOR K=1 TO 10
▲ 30 READ I$(K)
  40 NEXT K
  50 PRINT «ESCOGE UN NUMERO DEL 1 AL 10»
  60 INPUT N
  70 PRINT «POR QUE NO »;
  80 PRINT I$(N)
  90 PRINT «O.K. »;
 100 INPUT A$
 110 IF A$=«SI» THEN STOP
 120 PRINT:GOTO 50
 130 DATA PINTAS UN CUADRO,LEES UN LIBRO
```

Añade tus datos en el paso 130.

Cambios para Sinclair (Timex)

```
  10 DIM I$(10,?)
▲ 30 INPUT I $(K)
```

Cuenta cuántos caracteres tiene tu cadena más larga y coloca la cifra en la instrucción DIM.

Tabla de números aleatorios

```
100 FOR K=1 TO 10
110 PRINT K;TAB(4);
120 FOR L=1 TO A(K)
130 PRINT «*»
140 NEXT L
150 PRINT
160 NEXT K
```

Escribir subrutinas (página 26)

Encuesta sobre helados

```
△ 10 DIM I$(5),N(5)
  20 FOR K=1 TO 5
▲ 30 READ I $(K), N(K)
  40 GOSUB 70
  50 NEXT K
  60 STOP
  70 PRINT I $(K);TAB(11);« »;
  80 FOR L=1 TO N(K)
  90 PRINT «*»;
 100 NEXT L
 110 PRINT
 120 RETURN
 130 DATA MELON,16,PLATANO,11
 140 DATA JENJIBRE,8,PEPINILLOS,1
 150 DATA CHICLE,18
```

Necesitas la instrucción TAB en el paso 70 para estar seguro que la computadora dibuja los asteriscos a la misma distancia en cada línea.

Cambios para Sinclair (Timex)

```
  10 DIM I$(5,10)
  15 DIM N(5)
▲ 30 INPUT I $(K)
▲ 35 INPUT N(K)
```

Hunde el submarino

```
200 PRINT «FALLASTE»
210 PRINT «DISPARA AL »;
220 IF B=Y THEN GOTO 250
230 IF B<Y THEN PRINT «NORTE»;
240 IF B>Y THEN PRINT «SUR»;
250 IF A=X THEN GOTO 280
260 IF A<X THEN PRINT «ESTE»
270 IF A>X THEN PRINT «OESTE»
280 RETURN
```

Esta es la subrutina para comparar la situación del submarino con las coordenadas que da el jugador y escribir el correspondiente mensaje en la pantalla.

Veinte preguntas

Si tus respuestas incluyen símbolos o letras, alinéalos en orden.

```
△ 10 DIM Q$(20) ,A(20)
  20 FOR K=1 TO 20
▲ 30 READ Q$(K) ,A(K)
  40 NEXT K
  50 FOR L=1 TO 20

  60 PRINT Q$(L)
  70 INPUT X
  80 IF X=A(L) THEN PRINT «CORRECTO»
  90 IF X<>A(L) THEN PRINT «NO TONTO.
LA RESPUESTA ES »;A(L)
 100 PRINT
 110 NEXT L
 120 DATA ¿CUANTAS PATAS TIENE UN CIEMPIES?,100
 130 DATA ¿CUANTOS JUGADORES HAY EN UN EQUIPO DE FUTBOL?,1
```

Pon tus respuestas y preguntas en pasos data como éstos.

Cambios para Sinclair (Timex)

```
  10 DIM Q$(20,?)
  15 DIM A(20)
▲ 30 INPUT Q$(K)
▲ 35 INPUT A(K)
```

Cuenta el número de caracteres de tu cadena más larga y pon la cifra en la instrucción DIM.

Programa para una máquina tragaperras

```
△ 10 DIM F$(6)
  20 FOR K=1 TO 6
▲ 30 READ F$(K)
  40 NEXT K
  50 CLS
  60 LET T=10
  70 PRINT «TIENES »;T;« FICHAS»
  80 LET I $=INKEY$
  90 IF I $=«» THEN GOTO 80
 100 LET T=T-1
 110 CLS
 120 LET R=INT(RND(1)*6+1)
 130 LET A$=F$(R)
 140 LET R=INT(RND(1)*6+1)
 150 LET B$=F$(R)
 160 LET R=INT(RND(1)*6+1)
 170 LET C$=F$(R)
 180 PRINT
 190 PRINT A$;« »;B$;« »;C$
 200 PRINT
 210 IF A$=B$ AND B$=C$ AND
C$=«CEREZA» THEN GOSUB 270
 220 IF A$=B$ AND B$=C$ AND
THEN GOSUB 310
C$<>«CEREZA»
 230 IF (A$=B$ AND B$<>C$) OR
(A$=C$ AND C$<>B$) OR (B$=C$
AND C$<>A$) THEN GOSUB 350
 240 IF T>0 THEN GOTO 70
 250 PRINT «NO TE QUEDAN FICHAS»
 260 STOP
 270 PRINT «3 CEREZAS»
 280 PRINT «GANAS EL ESPECIAL»
 290 LET T=T+20
 300 RETURN
 310 PRINT «3 DE CUALQUIER CLASE»
 320 PRINT «GANAS 5 FICHAS»
 330 LET T=T+5
 340 RETURN
 350 PRINT «2 DE CUALQUIER CLASE»
 360 PRINT «GANAS 2 FICHAS»
 370 LET T=T+2
 380 RETURN
 390 DATA LIMON,CEREZA,MELON
 400 CAMPANA,UVAS,CIRUELAS
```

Selecciona tres frutos de F$.

Escribe las frutas.

Los pasos 210-230 calculan si has ganado.

1.ª subrutina.

2.ª subrutina.

3.ª subrutina.

Cambios para Sinclair (Timex)

```
  10 DIM F$(6,6)
▲ 20 INPUT F$(K)
```

La caza del tesoro (páginas 28-33)

Este es el programa completo para la caza del tesoro listado sección a sección en el orden correcto. Los cambios para las computadoras Sinclair (Timex) los encontrarás al final del programa.

1 Crear matrices y leer los datos

```
△ 100 DIM N(7),E(7),S(7),O(7),D$(7),
  T$(7),T(7)
  110 FOR K=1 TO 7
▲ 120 READ N(K) ,E(K) ,S(K) ,O(K)
  130 NEXT K
  140 FOR K=1 TO 7
▲ 150 READ D$(K)
  160 NEXT K
  170 FOR K=1 TO 7
▲ 180 READ T$(K) ,T(K)
  190 NEXT K
  200 LET M=0
  210 LET C=0
  220 LET F=0
  230 LET O=0
  240 LET X=0
  250 LET Y=0
  260 CLS
```

Ver paso 2000 para estos data.

2 Subrutina ayuda

```
  300 GOSUB 1000
```

Ver la subrutina en el paso 1000

3 Selección aleatoria de una habitación

```
  350 LET R=INT(RND(1)*7+1)
```

4 Identificar la habitación

```
  400 PRINT «ESTAS EN LA HABITACION »;R
  410 PRINT «ES »;D$(R)
  420 PRINT «CONTIENE:»;
```

Identificar los contenidos

```
  430 FOR K=1 TO 7
▲ 440 IF T(K)=R THEN PRINT TAB(15);
  T$(K):LET F=1
  450 NEXT K
  460 IF F=0 THEN PRINT TAB(15)«NADA»
  470 LET F=0
▲ 480 PRINT:PRINT
```

F es una variable bandera. En el paso 440, la bandera «sube» (F es 1) cuando uno de los números de la matriz T es igual al número de habitación (R).

5 Entrada (input) del jugador

```
  500 PRINT «QUE QUIERES HACER »
  510 INPUT A$
  520 IF A$="«AYUDA» THEN GOSUB 1000
  530 IF A$="«N» OR A$="«E» OR A$=«S» OR A$="«O»
  THEN GOSUB 1200
  540 IF A$="«COGER» THEN GOSUB 1300
  550 IF A$="«DEJAR» THEN GOSUB 1400
  560 IF A$="«POSICION» THEN GOSUB 1500
```

Los pasos 520 a 560 mandan a la computadora a una determinada subrutina dependiendo de la palabra que el jugador escriba en la variable A$ (paso 510).

10 Cuentan las jugadas

```
  600 LET M=M+1
```

11 ¿Están todos los tesoros en la misma habitación?

```
  610 LET W=T(1)
  620 FOR K=2 TO 7
  630 IF W<>T(K) THEN LET F=1
  640 NEXT K
  650 IF F=1 THEN GOTO 690
  660 PRINT «BIEN HECHO, TIENES TODOS
  LOS TESOROS»
  670 PRINT «EN LA HABITACION »;R;« EN »;M;«
  JUGADAS»
  680 STOP
  690 LET F=0
```

En el paso 610, el número almacenado en T(1) se introduce en la variable W. Dentro del bucle (paso 630) la computadora compara W con todos los números almacenados en la matriz T. Si cualquier número T es diferente a W, la computadora hace que la variable de bandera (F) valga 1. Todos los números de la matriz T coinciden con W F conserva el valor 0 que significa que todos los tesoros están en una misma habitación.

12 ¿Ha gastado el jugador todas sus jugadas?

```
  700 IF M<=28 THEN GOTO 730
  710 PRINT «LO SIENTO. SE TE HAN AGOTADO LAS
  JUGADAS»
  720 STOP
  730 PRINT
```

Si M vale más que 29, el jugador ha agotado todas sus jugadas, por lo que el programa para. Puedes cambiar el número de jugadas que el jugador puede hacer.

13 ¿Ha cambiado el jugador de posición?

```
  800 IF A$=«N» OR A$=«E» OR A$="«S» OR
  A$=«O» THEN GOTO 400
  810 PRINT «CONTINUAS EN LA HABITACION »;R
  850 GOTO 500
```

2 Subrutina ayuda

```
 1000 PRINT «HAY SIETE HABITACIONES EN
 ESTE LABERINTO»
 1010 PRINT «CON UNA  CAJA CON UN
 TESORO»
 1020 PRINT «EN CADA UNA. DEBES REUNIR
 TODOS»
 1030 PRINT «LOS TESOROS EN LA MISMA HABITACION»
 1040 PRINT
 1050 PRINT «ESTAS SON LAS INSTRUCCIONES QUE
 ENTIENDE LA COMPUTADORA»
 1060 PRINT «AYUDA        :TE DICE COMO DEBES
 JUGAR»
 1070 PRINT «N,E,S,O   :TE MUEVE AL NORTE,SUR,ESTE U OESTE»
 1080 PRINT «COGER         :TE PERMITE LLEVARTE EL TESORO»
 1090 PRINT «DEJAR         :DEJAS EL TESORO»
 1100 PRINT «POSICION    :MUESTRA LA POSICION
 ACTUAL DEL TESORO»

▲ 1110 PRINT:PRINT
 1120 RETURN
```

La computadora te dice cómo funciona el juego y las palabras que entiende. Cada vez que el jugador introduce la palabra HELP, la computadora acudirá a esta subrutina.

6 Subrutina mover

```
 1200 IF A$=«N» THEN LET X=N(R)
 1210 IF A$=«E» THEN LET X=E(R)
 1220 IF A$="«S» THEN LET X=S(R)
 1230 IF A$="«O» THEN LET X=O(R)
▲ 1240 IF X=0 THEN PRINT «IMPOSIBLE IR POR AHI»:GOTO 1260
 1250 LET R=X
 1260 RETURN
```

En los pasos 1200-1230, la computadora comprueba la dirección del jugador (N, E, S u O) y luego encuentra el número de habitación en esa dirección utilizando R como «número de compartimento» de la correspondiente matriz y lo almacena en X. Si X es 0 (paso 1240), quiere decir que no hay ninguna habitación en esa dirección.

7 Subrutina coger

```
▲1300 IF C=1 THEN PRINT «NO PUEDES LLEVAR
 MAS»:GOTO 1370
 1310 FOR K=1 TO 7
 1320 IF T(K)=R THEN LET Y=K
 1330 NEXT K
 1340 IF Y=0 THEN PRINT «ESTA HABITACION ESTA
 VACIA»
 1350 LET T(Y)=999
 1360 PRINT «OK LLEVAS:»; T$(Y)
▲1370 LET C=1: LET Y=0
 1380 RETURN
```

En el paso 1320, la computadora compara cada número almacenado en la matriz T con R. Si son iguales toma el número de compartimento de ese elemento de T y lo almacena en una variable Y. Después del bucle, la computadora usa Y para recoger el elemento correcto de T y cambia su valor a 999 (paso 1350). Si Y es 0 (paso 1340), significa que la habitación está vacía.

8 Subrutina dejar

```
▲1400 IF C=0 THEN PRINT «NO LLEVAS
 NADA»:GOTO 1450
 1410 FOR K=1 TO 7
 1420 IF T(K)=999 THEN PRINT T$(K);
 " DEJADO EN LA HABITACION »;R
 1430 IF T(K)=999 THEN LET T(K)=R
 1440 NEXT K
 1450 LET C=0
 1460 RETURN
```

La computadora encuentra el elemento de T que vale 999 (paso 1420) para cambiar su valor a R (paso 1430).

9 Subrutina Posición

```
 1500 PRINT «LLEVAS : »;
▲1510 IF C=0 THEN PRINT TAB(10);
 «NADA»:GOTO 1550
 1520 FOR K=1 TO 7
 1530 IF T(K)=999 THEN PRINT
 TAB(10);T$(K)
 1540 NEXT K
 1550 PRINT «CONTENIDO DE LAS HABITACIONES:»
 1560 FOR K=1 TO 7
 1570 IF T(K)<>999 THEN PRINT
 T(K);« : »;T$(K)
 1580 NEXT K
 1590 RETURN
```

1 Los data

```
▲2000 DATA 2,7,6,0
 2010 DATA 0,3,7,1
 2020 DATA 0,0,4,2
 2030 DATA 3,0,5,7,
 2040 DATA 7,4,0,6
 2050 DATA 1,5,0,0
 2060 DATA 2,4,5,1
```

Estos son los datos para las matrices N, S, E, O.

```
2100 DATA FRIA Y TENEBROSA,OSCURA
Y SOMBRIA
2110 DATA GRIS Y FANTASMAL,SUCIA
Y BRUMOSA
2120 DATA VACIA Y LUGUBRE,ENCANTADA
Y HORRIBLE,ATERRADORA Y MISTERIOSA
```

Estos son los datos para D$.

```
2200 DATA ORO,1,CHICLE,2
2210 DATA BOCADILLOS,3,BASURA,4
2220 JARROS DE MIEL,5,JOYAS,6,MONEDAS,7
```

Estos son los datos para T$ y T.

Cambios para Sinclair (Timex)

No hacen falta cambios para el Spectrum (Timex 2000) excepto en la manera de dimensionar las matrices, que ha de hacerse como en el ZX81 (Timex 1000). Cada instrucción DIM debe separarse por dos puntos o ponerse en un paso diferente como en el ZX81 (Timex 1000).

Paso 100: Coloca las instrucciones DIM en pasos separados y dimensiones D$ (7,20) y T$ (7,23).

Paso 120: Coloca pasos INPUT por separado para cada matriz y escribe los datos cuando ejecutes el programa.

```
120 INPUT N(K)
123 INPUT E(K)
125 INPUT S(K)
127 INPUT O(K)
```

Paso 150: Usa INPUT y escribe los datos cuando ejecutes el programa.

```
150 INPUT D$(K)
180 INPUT T$(K)
```

Paso 180: Usa INPUT para la matriz T$ y LET para la matriz T para que no tengas que introducir la posición inicial de los tesoros cada vez que ejecutes el programa. Utiliza un nuevo bucle en pasos 193-197 para la instrucción LET de la siguiente manera:

```
193 FOR K=1 TO 7
195 LET T(K)=K
197 NEXT K
```

Paso 350: Utiliza la instrucción RND de tu computadora.

```
350 LET R=INT(RND*7+1)
```

Paso 440: Utiliza pasos separados repitiendo la instrucción IF/THEN de la siguiente manera:

```
440 IF T(K)=R THEN PRINT TAB(15);T$(K)
450 IF T(K)=R THEN LET F=1
```

Pasos 480 y 1110: Utiliza pasos separados.

Pasos 1240-1510: Usa pasos separados y cuando haya una instrucción IF/THEN repítela en un nuevo paso. Ej.:

```
1240 IF X=0 THEN PRINT «IMPOSIBLE IR POR AHI»
1245 IF X=0 THEN GOTO 1260
```

Paso 2000: Suprime los pasos 2000-3000 y añade estos pasos para no tener que introducir los datos cada vez que ejecutas el programa. Cuando quieras volver a jugar, simplemente escribe GOTO 2000. (Cuando aprietas RUN la computadora borra todas las variables y matrices.)

```
2000 PRINT «¿QUIERES VOLVER A JUGAR?»
2010 INPUT R$
2020 IF R$="SI» THEN GOTO 193
2030 IF R$="NO» THEN STOP
2040 GOTO 2000
```

Instrucciones en BASIC

Esta es una lista de las palabras en BASIC usadas en este libro con breves explicaciones de lo que significa cada una. Las palabras señaladas con un asterisco no son iguales en todas las computadoras. En estos casos, busca en la tabla de conversión de la página 47 o en tu manual para saber cuál es la palabra que utiliza tu computadora.

ABS le dice a la computadora que ignore los signos positivo o negativo que encuentre delante de los números y que dé su valor absoluto. Por ejemplo, ABS(−5) es 5.

***ASC** convierte un carácter en su número de código ASCII correspondiente. Por ejemplo, ASC(«A») te dará 65, ya que éste es el número de código ASCII de la letra A. Las computadoras Sinclair usan CODE en lugar de ASC.

**CHR$** convierte un número en un carácter según los números de código que representen los caracteres en la computadora. En computadoras que utilizan el código ASCII, CHR$(65) da la letra A. En el ZX81 (Timex 1000), CHR$(38) te dará A, ya que esta computadora usa diferentes números de código.

CLS borra la pantalla.

***CODE** se utiliza en computadoras Sinclair en lugar de ASC para convertir una letra en su número de código. El Spectrum (Timex 2000) usa los números de código ASCII, mientras que el ZX81 (Timex 1000) utiliza sus propios números.

***DATA** ver READ/DATA.

***DIM** indica cuánto espacio de memoria hay que reservar para una matriz. Se escribe la palabra DIM seguida del nombre de la matriz y del número de elementos que contiene. Por ejemplo, DIM A(5) significa que hay cinco datos en la matriz A. Con las matrices de cadena en las computadoras Sinclair (Timex) debes poner el número de caracteres de la cadena más larga en la instrucción DIM.

FOR/TO ... NEXT hace que la computadora repita las instrucciones entre los pasos FOR/TO y NEXT un determinado número de veces. Esto se denomina un bucle.

GOSUB dice a la computadora que deje el programa principal y vaya a una sección del programa llamada subrutina. GOSUB debe ir seguido por el número del primer paso de la subrutina. Al final de la subrutina una instrucción RETURN le dice a la computadora que vuelva al programa principal comenzando en la instrucción posterior al paso GOSUB.

GOTO hace que la computadora salte al paso que se indica en el número que sigue a la palabra GOTO.

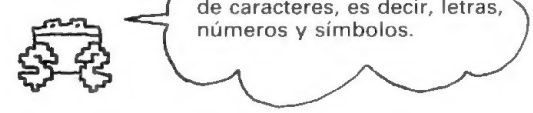
IF/THEN dice a la computadora que tome una decisión y haga algo según el resultado. Después de la palabra IF hay una condición que la computadora debe comprobar comparando datos. Si la condición es verdadera, la computadora seguirá con las instrucciones que sigan a la palabra THEN. Si la condición no es verdadera, las ignorará.

***INKEY$** inspecciona el teclado para ver si hay alguna tecla apretada. No espera a que la aprietes como INPUT, ni necesita de la tecla RETURN (o ENTER o NEWLINE según la computadora). Debido a que las computadoras trabajan tan deprisa, INKEY$ suele usarse dentro de un bucle. En algunas computadoras tienes que introducir un número entre paréntesis detrás de INKEY$ para indicarle a la computadora cuánto debe esperar.

INPUT le dice a la computadora que espere a que teclees información en una variable mientras se ejecuta el programa, INPUT debe ir seguido de un nombre de variable.

INT es la abreviatura de Integer (número entero). Transforma un número real (un número con decimales) en un número entero ignorando todo lo que hay detrás del punto decimal. Por ejemplo, INT(6.732) te dará 6. Con los números negativos hace lo mismo, pero redondeando hacia el siguiente número entero; es decir, INT(−3. 2) te dará −4.

***LEFT$** le dice a la computadora que tome un número de caracteres desde el extremo izquierdo de la cadena. Por ejemplo, LEFT$(A$, 4) le dice a la computadora que tome los primeros cuatro caracteres de A$. Las computadoras Sinclair (Timex) no utilizan esta instrucción.



LEN da la longitud de una cadena, es decir, el número de caracteres (incluyendo espacios y puntuación) de la cadena.

LET etiqueta los espacios de memoria y pasa alguna información en ellos.

LIST le dice a la computadora que exponga el listado del programa en la pantalla.

***MID$** le dice a la computadora que tome un determinado número de caracteres de una cadena. Por ejemplo: MID$(A$, 5, 2) toma dos caracteres de la variable A$ comenzando en la quinta letra. Las computadoras Sinclair (Timex) no utilizan esta instrucción.

NEXT le dice a la computadora que vuelva al principio del bucle. Ver FOR/TO ... NEXT.

NEW le dice a la computadora que borre el programa.

PRINT le dice a la computadora que exponga algo en la pantalla. PRINT por sí solo deja una línea en blanco.

***READ/DATA** le dice a la computadora que lea los datos listados en los pasos que comienzan con la palabra DATA y los ponga en la variable o matriz que sigue a la instrucción READ.

REM es la abreviatura de Remark (comentario) y se utiliza para recordar qué función tienen las diferentes partes del programa.

RESTORE le dice a la computadora que vuelva al principio de los pasos DATA.

RETURN se utiliza al final de una subrutina para decirle a la computadora que vuelva al programa principal comenzando en la instrucción posterior al paso GOSUB.

***RIGHT$** ordena a la computadora que tome un número de caracteres de la parte derecha de una cadena. Por ejemplo, RIGHT$(A$, 4) le dice a la computadora que tome los últimos cuatro caracteres de A$. Las computadoras Sinclair (Timex) no utilizan esta instrucción.

***RND** le dice a la computadora que tome un número aleatorio.

RUN le dice a la computadora que lleve a cabo las instrucciones de un programa.

***SPC** le dice a la computadora que deje un cierto número de espacios en la pantalla. No todas las computadoras tienen esta instrucción.

STEP se utiliza con bucles FOR/TO ... NEXT para decir a la computadora cuan a menudo debe repetir el bucle.

***TAB** mueve el cursor un número de espacios a lo largo de la pantalla. Suele usarse con PRINT para escribir algo en el centro de la pantalla. En algunas computadoras puedes utilizar TAB para mover el cursor un número determinado de líneas hacia abajo de la pantalla. En casi todas las computadoras necesitas poner un punto y coma detrás de TAB.

THEN se utiliza con IF para decirle a la computadora lo que debe hacer si ciertas condiciones se cumplen. Ver IF/THEN.

Símbolos BASIC

*	Multiplicar
/	Dividir
SQR	Hallar la raíz cuadrada
^	Elevar a. No es igual en todas las computadoras, así que busca en tu manual para ver cuál usa la tuya.
>	Mayor que
<	Menor que
>=	Mayor o igual que
<=	Menor o igual que
<>	No igual a

Tabla de conversión

Instrucciones usadas en este libro	CLS Borra la pantalla.	INT(RND(1)*N+1) Elije un número entero aleatorio entre 1 y N	INKEY$ Hace que la computadora espere a que aprietes una tecla.
BBC	CLS	RND(N)	INKEY$(N)
VIC 20/PET	PRINT CHR$(147)	INT(RND(1)*N+1)	GET X$
DRAGON	CLS	RND(N)	INKEY$
ORIC	CLS	INT(RND(1)*N+1)	KEY$
TRS 80	CLS	INT(RND(0)*N+1)	INKEY$
APPLE	HOME	INT(RND(1)*N+1)	X$=«» IF PEEK (-16384) >127 THEN GET X$
ZX81 TIMEX (1000)	CLS	INT(RND* N+1)	INKEY$
SPECTRUM TIMEX (2000)	CLS	INT(RND*N+1)	INKEY$

Conversiones adicionales para computadoras Sinclair (Timex)		
LEFT$ RIGHT$ MID$	Las computadoras Sinclair (Timex) no manejan las cadenas de la misma manera que las demás computadoras. Para convertir estas instrucciones debes contar la posición del primer y del último caracter que quieras seleccionar. Pon estas posiciones entre paréntesis con la palabra TO detrás de la cadena o de su nombre de variable tal y como se muestra en los ejemplos de la derecha.	LET X$=«ARDILLAS» PRINT X$(1 TO 4) ARDI PRINT X$(5 TO) LLAS PRINT X$(2 TO 4) RDI
ASC	Usa CODE en lugar de ASC. El ZX81 (Timex 1000) tiene sus propios números de código mientras el Spectrum (Timex 2000) usa el código ASCII.	PRINT CODE(«A»)
DIM	Para dimensionar una matriz de cadena utiliza DIM seguido de dos números entre paréntesis. El primero será el número de caracteres de la cadena más larga. En el ejemplo de la derecha, la matriz A$ contiene cinco cadenas, la mayor de las cuales tiene 12 caracteres.	DIM A$(5,12)
READ/DATA	El Spectrum (Timex 2000) utiliza READ/DATA como las demás computadoras; no así el ZX81 (Timex 1000), que no tiene ningún equivalente parecido. La mejor manera de sustituirlo es mediante una serie de instrucciones INPUT como las que se muestran a la derecha.	10 FOR K=1 TO 5 20 INPUT A$(K) 30 NEXT K

Indice

En este índice encontrarás los nombres de los problemas, las páginas en las que se encuentran las soluciones de los problemas están señaladas en negrita.

Printed in Spain. Impreso en España por MELSA
Depósito Legal M–34849–1986
I.S.B.N. 84–7374–159–5